UNIVERSIDAD INDUSTRIAL DE SANTANDER
ESCUELA DE INGENIERÍAS ELÉCTRICA, ELECTRÓNICA Y DE TELECOMUNICACIONES
Perfecta Combinación entre Energía e Intelecto

INCLINÓMETRO BASADO EN UN MICROCONTROLADOR BASIC STAMP II USANDO ACELEROMETROS DE TECNOLOGÍA MEMS (MICRO ELETRO-MECHANICAL SYSTEMS)

EFREN DARIO DUGARTE MATEUS 1962201

HUGO FABIAN MENDEZ SALAZAR 1972646

Presentado ante:
Consejo de Escuela de Ingenierías Eléctrica, Electrónica y de Telecomunicaciones, Facultad de Ingenierías Físico-Mecánicas.

UNIVERSIDAD INDUSTRIAL DE SANTANDER

FACULTAD DE INGENIERÍAS FÍSICO-MECÁNICAS

ESCUELA DE INGENIERÍAS ELÉCTRICA, ELECTRÓNICA Y DE TELECOMUNICACIONES

GRUPO DE INVESTIGACIÓN "*CEMOS*"

BUCARAMANGA
2004

INCLINÓMETRO BASADO EN UN MICROCONTROLADOR BASIC STAMP II USANDO ACELEROMETROS DE TECNOLOGÍA MEMS (MICRO ELETRO-MECHANICAL SYSTEMS)

EFREN DARIO DUGARTE MATEUS

HUGO FABIAN MENDEZ SALAZAR

Trabajo para optar el título de
INGENIERO ELECTRÓNICO

Director:
ING. JOSÉ ALEJANDRO AMAYA PALACIO

UNIVERSIDAD INDUSTRIAL DE SANTANDER

FACULTAD DE INGENIERÍAS FÍSICO-MECÁNICAS

ESCUELA DE INGENIERÍAS ELÉCTRICA, ELECTRÓNICA Y DE TELECOMUNICACIONES

GRUPO DE INVESTIGACIÓN "*CEMOS*"

BUCARAMANGA
2004

DEDICATORIA

A mi abuela, quien fue mi apoyo e inspiración en todo momento, a mis hermanas quienes siempre creyeron en mí y al tío Hugo ejemplo de lucha y constancia.

FABIAN MENDEZ S.

A mis padres, por creer en mí, por su paciencia y confianza incondicional.

EFREN DUGARTE M.

AGRADECIMIENTOS

Queremos agradecer al profesor José Amaya por haber creído en nosotros y por apoyarnos en los momentos más difíciles. A William Acevedo por toda su colaboración, a los Ingenieros Omar Peña y Jairo Cala por sus recomendaciones.

CONTENIDO

Pag.

LISTA DE FIGURAS

Pag.

LISTA DE TABLAS

Pag.

LISTA DE ANEXOS

Pag.

RESUMEN

TITULO: INCLINÓMETRO BASADO EN UN MICROCONTROLADOR BASIC STAMP II USANDO ACELEROMETROS DE TECNOLOGÍA MEMS (MICRO ELETRO-MECHANICAL SYSTEMS)[*].

AUTORES: DUGARTE MATEUS, Efrén Darío[**]
MÉNDEZ SALAZAR, Hugo Fabián[**]

PALABRAS CLAVES: Inclinación, acelerómetro, Basic Stamp 2, Pbasic, Labview.

DESCRIPCIÓN:

El inclinómetro es una herramienta de medición de ángulos de inclinación en dos ejes (para el rango entre 60° y -60°) el cual esta conformado por un modulo o panel de control y una pequeña caja donde se aloja el sensor, conectados entre si por medio de un cable blindado de cuatro hilos de aproximadamente 50 centímetros de longitud.

La circuitería se puede dividir en dos etapas: la correspondiente a la etapa de adquisición, cuyo componente principal es el acelerómetro, y la otra etapa que tiene que ver con el procesamiento (microcontrolador Basic Stamp 2), almacenamiento, visualización y descarga de datos, en la cual se encuentra el microcontrolador, la memoria externa, el LCD y el puerto serial DB9.

La batería utilizada es de +9V, suficiente para 150 horas de operación. Para reducir el consumo de potencia del equipo en el modo de alimentación con batería se suprime la utilización del regulador LM7809, ya que no se es de utilidad, debido a que la batería no suministra más de 9 voltios, de manera que, solo se utiliza el LM7805 para regular el voltaje de alimentación del LCD y la memoria externa. Esto conduce a tener ahorro en el consumo de potencia.

El cable de comunicación entre inclinómetro y PC tiene características de comunicación serial, además de tener un blindaje para reducir posibles ruidos externos.

El único software que es necesario para descargar los datos al PC es un programa de comunicación serial (Labwiev).

[*] Proyecto de Grado
[**] Facultad de ciencias Físico Mecánicas, Escuela de Ingenierías Eléctrica Electrónica y Telecomunicaciones. José Alejandro Amaya Palacio

ABSTRACT

TITLE: TILT SENSOR BASED ON MICROCONTROLLER BASIC STAMP II USING ACCELEROMETER OF TECHNOLOGY MEMS MICRO ELETRO-MECHANICAL SYSTEMS)[*].

AUTHOR: DUGARTE MATEUS, Efrén Darío[**]
MÉNDEZ SALAZAR, Hugo Fabián[**]

KEY WORKS: Tilt, accelerometer, Basic Stamp 2, Pbasic, Labview.

DESCRIPTION:

The tilt sensor is a tool of mensuration of angles of inclination in two axes (for the range between 60° and -60°) which is conformed for an module or control panel and a small box where the sensor is lodged, connected among if by an armored cable of four threads of approximately 50 centimeters of longitude.

The circuitry can be divide in two stages: the corresponding to the stage of acquisition whose main component is the accelerometer, and the other stage that has to do with the processing (Microcontrollers Basic Stamp 2), storage, visualization and download of data, in which is the microcontroller, the external memory, the LCD and the serial port DB9.

The used battery is of +9V, enough for 150 hours of operation. To reduce the consumption of power of the prototype in the feeding way with battery the use of the regulator LM7809 is suppressed, since it is not been of utility, because the battery doesn't give more than 9 volts, so that, just the LM7805 is used to regulate the voltage of feeding of the LCD and the external memory. This leads to have saving in the consumption of power.

The communication cable between tilt sensor and PC has characteristic of serial communication, besides having a shielding to reduce possible external noises.

The only software that is necessary to discharge the data to the PC is a program of serial communication (Labview).

[*] Proyect of Grade.
[**] Faculty of Sciences Physique Mechanics. School of engineerings Electric, Electronic and Telecommunications. José Alejandro Amaya Palacio

INTRODUCCIÓN

El problema de medir ángulos de inclinación es común en campos como la Física, Ing. Civil, Ing. Metalúrgica y en aplicaciones avanzadas como la Robótica, donde estos valores de inclinación pueden ser usados como información para determinar algún otro parámetro, definir características de una estructura o para utilizarla como una variable de retroalimentación para la corrección de posición de algún mecanismo. La necesidad de obtener y procesar estas variables físicas, generó el interés del grupo de investigación CEMOS[*] en presentar nuevas opciones de tecnología, desarrollando aplicaciones que realicen estas tareas. Estas aplicaciones contribuyen al fortalecimiento del área de instrumentación electrónica.

Con el propósito de presentar un nuevo dispositivo programable en el ámbito de la E3T (Escuela de Ingenierías Eléctrica, Electrónica y Telecomunicaciones), se diseña e implementa una herramienta para la toma de medidas de inclinación basada en un dispositivo perteneciente a la familia de microcontroladores Basic Stamp, los cuales han sido introducidos en el mercado hace ya varios años por la empresa Norteamericana PARALLAX. Este dispositivo microcontrolador es particular debido a su lenguaje de programación “PBASIC”, el cual es muy similar al lenguaje Basic. Además, se busca ampliar los conocimientos relacionados con el uso de los acelerómetros, como su forma de calibración, limitación de ancho de banda, determinación del periodo de la señal de salida e interpretación de las variaciones de la aceleración en unidades de gravedad.

[*] Control, electrónica, modelamiento y simulación

Con la presentación de este microcontrolador se busca tener una alternativa diferente a los ya conocidos PIC y Motorota al momento de decidir trabajar en una aplicación que requiera el uso de un dispositivo programable (microcontroladores). El Basic Stamp 2 es un hibrido que cuenta con dispositivos internos como un chip interprete (PIC16C57), un oscilador de 20 MHz, una memoria EEPROM para almacenamiento de programa, un regulador LM2936 de 5 voltios para alimentar los demás dispositivos internos y una configuración para manejo de puerto serial. Todo esto con el fin de reducir la construcción de un circuito impreso, y así disminuir el tiempo de diseño.

En el capítulo 1 se realiza una descripción detallada de las características del sensor utilizado para la toma de medidas de inclinación. Además se realiza una breve discusión acerca del porqué se optó por utilizar como censor un acelerómetro, fundamentados en las propiedades del mismo y que cumplen con las prestaciones necesarias para esta aplicación. Otro aspecto importante que se aborda es la forma como el acelerómetro mide inclinación a partir de variaciones en la aceleración y como se realiza su calibración.

En el capítulo 2 se presentan las características del inclinómetro, su aspecto externo y la forma de manejo del menú. Dentro de este capítulo se realiza una descripción detallada de la forma de alimentación, la comunicación y descarga de los datos además de las diferentes formas utilizadas para la visualización de las medidas de inclinación. Además se presenta el plano completo del inclinómetro donde se puede observar la interconexión entre los dispositivos que lo componen.

En el capítulo 3 se describen los diferentes dispositivos que conforman el inclinómetro, dando una descripción detallada acerca del microcontrolador entre las cuales se encuentran sus características, la descripción de cada uno de sus pines, el lenguaje y la conexión correcta para realizar su programación, además

de unas precauciones que se recomiendan tener en cuenta al momento de trabajar con este microcontrolador. Se presentan los impresos que conforman el inclinómetro, junto a los criterios que se tuvieron en cuenta al momento de su diseño.

En el capítulo 4 se presenta el software del microcontrolador que rige todas las funciones del inclinómetro, las instrucciones utilizadas junto a una explicación de donde y para que se utilizaron. Además se realiza una explicación de cada una de las subrutinas que conforman el software mediante una representación en diagramas de bloques con su respectiva explicación.

En el capítulo 5 se consignan algunos de los resultados obtenidos a lo largo del desarrollo del inclinómetro, entre los que se encuentra el consumo de potencia, y los resultados de las pruebas realizadas con el inclinómetro.

En el capítulo 6 se enumeran las conclusiones obtenidas a lo largo del desarrollo del proyecto. Se hacen algunas recomendaciones para futuras aplicaciones del inclinómetro, además de posibles mejoras del mismo.

En el capítulo 7 se dan algunas sugerencias para el uso y posible mejoramiento del proyecto.

En los anexos se presentan las hojas de datos del LCD, la memoria y un resumen de la arquitectura del microcontrolador con la cual se pretende proporcionar una herramienta que ayude a la comprensión del funcionamiento del microcontrolador.

1. ACELERÓMETRO

Cuando se conduce un vehículo se experimenta la aceleración cuando nuestro cuerpo se hunde en el asiento; en la medida en que el automóvil y el asiento adquieren mayor velocidad (acelerándose), se percibe una deformación en el asiento proporcional a la intensidad de la aceleración, Esto se debe a que el cuerpo del conductor posee inercia y tiende a mantener su velocidad inicial. Si fuese posible medir el grado de deformación o inclinación, se tendría un medidor de aceleración.

La medición de aceleración es importante en situaciones que involucran cualquier tipo de sistema físico. Los acelerómetros (medidores de aceleración) son dispositivos cuyo voltaje instantáneo de salida es proporcional al valor instantáneo de la aceleración. Estos son utilizados actualmente en casi todas las mediciones de vibración, choques e impactos.

La mayor dificultad encontrada clásicamente en el diseño de instrumentos para llevar a cabo estas mediciones de aceleración, ha sido siempre el elevado costo de los sensores y el tamaño de estos. Esta dificultad ha sido superada, al menos para el sector de aplicaciones de mediana precisión, mediante la introducción de los acelerómetros de tecnología **MEMS**[*].

Para el caso de medición de ángulos de inclinación (presentado en este libro), el acelerómetro es el elemento principal del circuito de la etapa de adquisición de datos; esta etapa consta de un acelerómetro **ADXL202E**[**] de tecnología **MEMS**, una resistencia de 10MOhm para determinar el periodo de las señales PWM y dos condensadores de 0.47 uF cada uno para una etapa de filtrado y limitación del

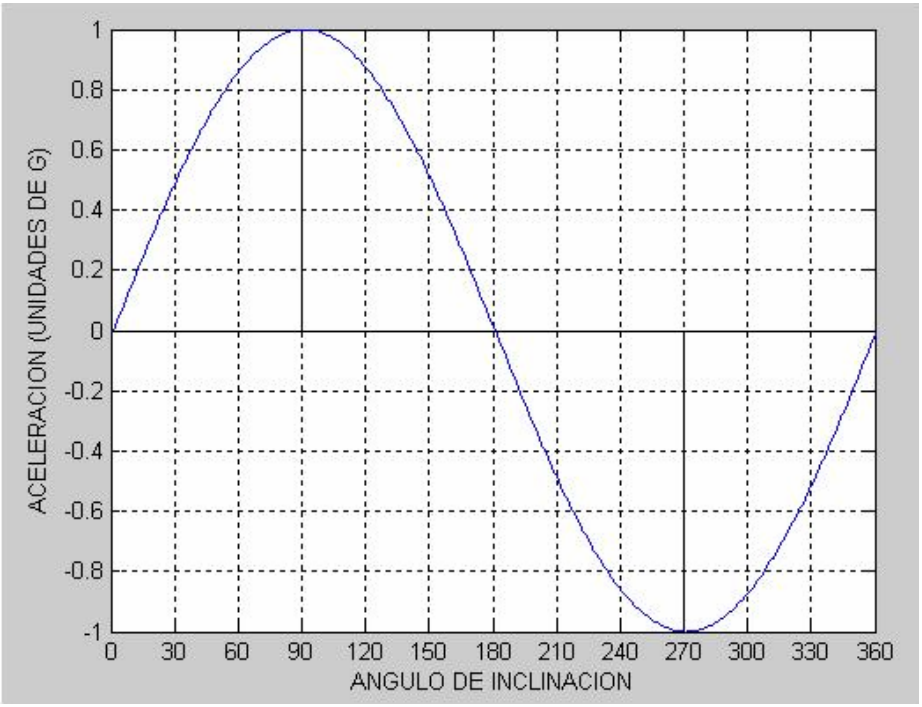
[*] Micro Eletro-Mechanical Systems.
[**] Ver numeral 1.2.

ancho de banda de la señal análoga del acelerómetro (Señal antes de pasar por el conversor A/D interno del acelerómetro)[**].

1.1 DETERMINACIÓN DE LA INCLINACIÓN A PARTIR DE LA ACELERACIÓN

La salida de un acelerómetro cuando permanece estacionario, mide la gravedad ejercida en él por la Tierra. Cuando un acelerómetro rota 360º, se origina una señal de salida sinusoidal del valor medido de aceleración en unidades de gravedad (*G*). Lo que significa que para determinar el valor del ángulo de inclinación es suficiente hallar el seno inverso del valor medido en *G* (Ver figura 1).

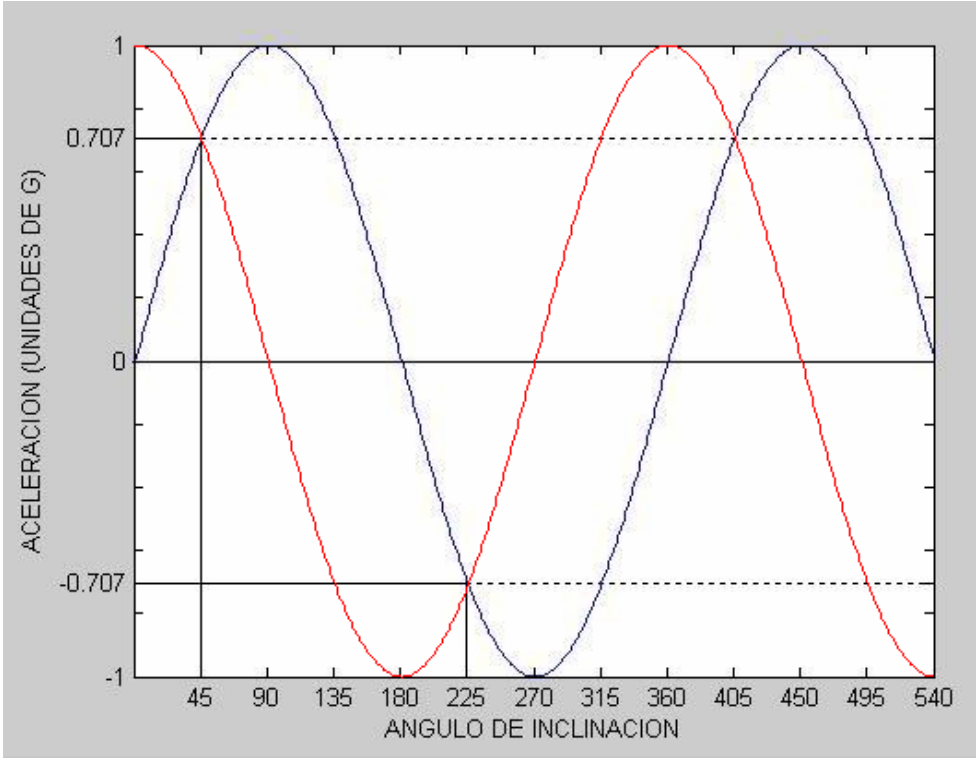
Figura 1. Señal de salida de un acelerómetro con un eje



El acelerómetro ADXL202E tiene dos ejes perpendiculares efectivos, por lo tanto las señales generadas corresponden al seno y coseno del ángulo de inclinación respectivo (Ver figura 2).

[**] Referirse al Anexo B.

Figura 2. Señal de salida en un acelerómetro de 2 ejes perpendiculares (ADXL202E)



De acuerdo con las figura 1 y 2, la relación entre la señal de salida del acelerómetro en unidades de aceleración (*G*) y la gravedad *G* esta dada por:

$$\begin{aligned} Ax &= G \bullet sen(\alpha) \\ Ay &= G \bullet sen(\beta) \end{aligned} \qquad (1)$$

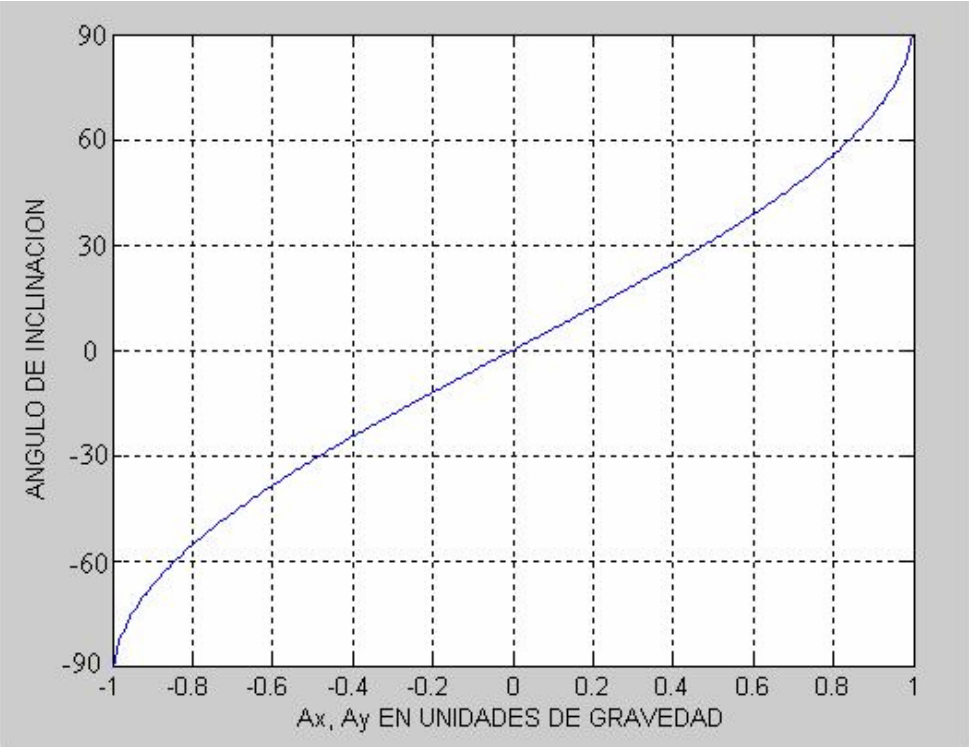
Donde Ax y Ay representan la señal de salida de cada uno de los ejes de referencia en el acelerómetro, *G* es el valor de aceleración debido a la gravedad, y α y β son los ángulos de inclinación.

Por lo tanto para calcular la inclinación es necesario determinar el valor del seno inverso (Arcsen):

$$\alpha = sen^{-1}(Ax/G)$$
$$\beta = sen^{-1}(Ay/G) \qquad (2)$$

A continuación se presenta la grafica para la función Arcsen.

Figura 3. Relación entre el ángulo de inclinación y la aceleración.



Como se puede observar en la figura 3, cuando el ángulo de inclinación es menor a 40º aproximadamente, la función Arcsen se puede representar por medio de una función lineal. Otra observación importante de la figura 3 es que a medida que el ángulo se acerca a los 90º, esta aproximación lineal ya no es una representación valida debido a que las variaciones en la salida del acelerómetro son cada vez menos significativas.

Debido a que el microcontrolador no es capaz de ejecutar una función arc sin, fue necesario realizar una interpolación con los intervalos mostrados en la tabla 1. En

esta tabla se puede observar que a medida que el ángulo se aproxima a 90° el delta entre los valores del seno se hacen cada vez más pequeños, lo que equivaldría a tener una menor resolución en caso de mantener intervalos de larga duración.

Tabla 1. Intervalos de interpolación:

Sin	Arc sin	Delta sin	Delta sin/deg.
0.000	00	-	-
0.342	20	0.342	0.017
0.643	40	0.301	0.015
0.820	55	0.177	0.012
0.906	65	0.086	0.009
0.940	70	0.034	0.006
0.966	75	0.026	0.005
0.985	80	0.019	0.004
0.996	85	0.011	0.002
1.000	90	0.004	0.001

Nota: deg =arc sin 2 – arc sin 1

Para aumentar la resolución del inclinómetro, los intervalos de interpolación se reducen, observe que los dos primeros intervalos (0º a 20º y 20º a 40º) se toman cada 20 grados y luego se reducen a 5 grados, a partir de 55º.

Para determinar la aceleración se utiliza la ecuación 4, pero debido a que el microcontrolador interpreta las variaciones de ciclo útil en unidades de 2us, la fórmula se transforma en una función equivalente, es decir:

$$\text{Aceleración} = [\text{TimeX-TCenterX}] / \text{Sensitividad} \qquad (3)$$

Donde:
TimeX = Valor medido por el acelerómetro en unidades de 2us
TCenterX = Valor medido en 0G por el acelerómetro en unidades de 2us
Sensitividad = Sensitividad medida en unidades de 2us (Ver tabla 3).

Dentro del software del inclinómetro esta ecuación fue desarrollada por partes, primero se halla el valor absoluto de la diferencia del numerador de la ecuación 3 para no tener en cuenta signos. En la segunda parte se realiza la división entre el resultado obtenido y la sensitividad, para evitar la perdida de muestras el residuo es multiplicado por 10 y sumado al resultado de la división.

1.2 ELECCIÓN DEL ACELERÓMETRO

Puesto que, para el caso de medición de inclinación, es necesario realizar mediciones de aceleración estática, los acelerómetros capacitivos son la mejor alternativa, pues cumplen con requisitos técnicos validos tales como: rangos de medida apropiados y alta sensibilidad a pequeñas variaciones de aceleración; además sus características de alta relación señal a ruido, su excelente estabilidad térmica e inmunidad contra campos magnéticos y eléctricos (EMI) los hacen ideales para aplicaciones donde se requiere una frecuencia relativamente baja.

Dentro de las diferentes clases de sensores capacitivos se encuentran los acelerómetros de tecnología **MEMS**. Esta tecnología crea dispositivos diminutos que combinan componentes mecánicos y eléctricos, estos dispositivos pueden tener dimensiones entre unos pocos micrómetros o milímetros. Empresas como ***ANALOG DEVICES***, ***PCBpiezotronics, Kistler Instrument Corporation, VTI Technologies,*** y ***Endevco Corporation,*** son algunas de las empresas fabricantes de estos acelerómetros. Algunos de los acelerómetros que ofrecen estas

empresas fabricantes como los proporcionados por ***Endevco Corporation*** y ***Kistler Instrument Corporation,*** fueron descartados debido a su elevado costo y difícil acceso. La empresa ***PCBpiezotronics*** ofrece el acelerómetro 3701 y 3702 los cuales cumplen con los requerimientos necesarios para la construcción del inclinómetro, pero sus rangos de medida están entre ±3G y ±200G. Debido a que el rango en que trabaja el inclinómetro es de ±1G, se descartaron estas opciones. Los acelerómetros de ***VTI Technologies*** proporcionan señales de salida análogas, lo que implicaría un el uso de un conversor A/D, por esto se descarto esta opción.

La empresa ***ANALOG DEVICES*** proporciona varias clases de acelerómetros de tecnología **MEMS**, los cuales trabajan con referencia al vector gravedad y unos rangos de medida de la aceleración en unidades de gravedad. Entre los que se encuentran el **ADXL190**, **ADXL202E**, y **ADXL210E**. El **ADXL190** proporciona un amplio rango de medida de aceleración de +/-100G (G=aceleración debida a la gravedad de la Tierra=9.807m/s^2) y señales de salida análogas, esto implicaría la implementación de una etapa de acondicionamiento de señal (un filtro, un amplificador y un conversor A/D), la cual aumenta los niveles de ruido de cuantización y el tiempo de procesamiento de la señal, por lo tanto se descarta la implementación del este acelerómetro.

El **ADXL210E** a pesar de proporcionar señales digitales cuyos ciclos útiles (relación del ancho de pulso con respecto al periodo) son proporcionales a la aceleración, presenta un rango de medida de +/-10G. Debido a que el rango de medida que se desea utilizar es +/-1G se optó por otra alternativa.

El acelerómetro seleccionado para la aplicación del inclinómetro fue el **ADXL202E** debido a que su señal de salida en cada uno de los dos ejes es digital (PWM) y su rango de medida es +/-2G (El rango que se necesita esta entre ±1G), con lo cual se disminuyen los niveles de ruido y se consigue un rango de medida apropiado.

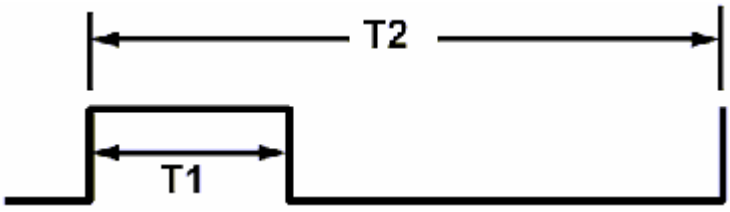
1.3 CARACTERÍSTICAS DEL ACELERÓMETRO ADXL202E

El acelerómetro ADXL202E de ***Analog Devices*** tiene bajo consumo de potencia, diseñado para medir bajos niveles de aceleración, además de poseer dos ejes de medida en un solo empaque; requiere un voltaje de alimentación entre 3V y 5,25V, y las salidas son de ciclo útil modulado, es decir PWM. Usando un pin del microcontrolador (Basic Stamp 2 es el microcontrolador utilizado en la construcción del inclinómetro), es posible determinar el ancho de los pulsos y así evitar una etapa de conversión A/D entre el sensor y el microcontrolador. Adicionalmente se puede usar otro pin para alimentar el sensor, y así reducir el uso de alguna fuente de alimentación adicional. La corriente estática requerida tiene un máximo de 1mA. En la aplicación presentada, el acelerómetro fue alimentado a 5 voltios por medio de un pin de salida en alto que brinda el microcontrolador. Algo para resaltar es la opción existente de alimentación en modo de consumo de baja potencia (alimentación a 3 voltios) la cual se descarta, ya que con ella se necesita una fuente auxiliar de alimentación o acondicionamiento de señal, lo que aumenta el desarrollo de hardware. Para el modo de conexión a baja potencia, el periodo de la señal PWM proveniente del acelerómetro debe tener un periodo de 10ms, lo cual reduce al tiempo de duración del ciclo útil de la señal y por consiguiente las variaciones en este, de tal forma que el número de muestras tomadas por el microcontrolador sobre la señal se ven disminuidas repercutiendo en una disminución de resolución (la rata de muestreo es de 2us), aunque cabe anotar que esta resolución es suficiente para la implementación realizada. Para mayor información de cómo utilizar el acelerómetro en modo de bajo consumo, refiérase al Anexo B.

En la Figura 4 se muestra la forma de onda de la señal de salida del acelerómetro, señal de ancho de pulso modulado (PWM) la cual es leída e interpretada por el

microcontrolador, relacionando la duración del ancho del pulso con la variación en la inclinación, por medio de la formula 4[*].

Figura 4. Relación entre aceleración y ciclo útil.



Fuente: Accelerometer memsic and Basic Stamp.

$$A(G) = ((T1/T2) - 0.5)/12.5\% \quad (4)$$

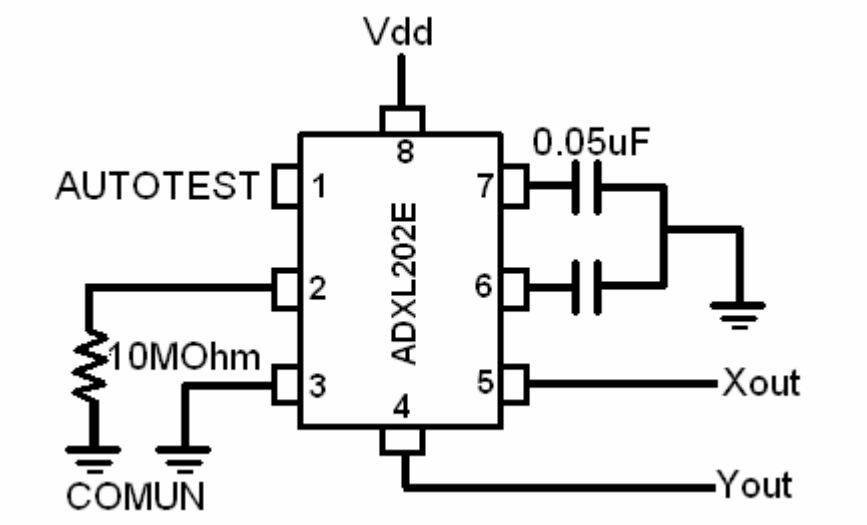
Donde:

A (G) = Aceleración en unidades de gravedad.
T1 = Ciclo útil se la señal.
T2 = Periodo de la señal.
0.5 = Idealmente el ancho del pulso es la mitad del periodo en 0*G.*
12.5 = Sensitividad ideal o factor de escala (Idealmente el ancho de pulso varía 12.5% del ciclo útil por cada *G*).

Las señales de ancho de pulso variable (PWM) son puestas en ambos canales[*] por la señal del resistor conectado entre el pin 2 del acelerómetro y tierra. El valor para esta resistencia determina el periodo de la señal de salida, a continuación en la Figura 5 se muestra el plano para la conexión del acelerómetro.

[*] ANALOG DEVICES, "ADXL202E Low-Cost ±2 G Dual-Axis Accelerometer with Duty Cycle Output [PDF]",
[*] Canal para el eje X y para el eje y.

Figura 5. Plano de conexión del acelerómetro.



La ecuación para el periodo de la señal esta dado por*:

$$T(S)=Rset\ \Omega\ /125\ M\Omega \qquad (5)$$

Se escogió un valor para Rset de 10MΩ, la cual determina un periodo en teoría de 80ms (T = 80 ms).

El ancho de banda del filtro esta puesto a 10Hz (Cx = Cy = 0.47 uF) para reducir la señal de ruido. Inicialmente se utilizaron condensadores de 0.05uF para un ancho de banda de 100Hz*, lo cual introducía un ruido significativo y la resolución era baja. Teniendo en cuenta la especificación proporcionada por el fabricante en la hoja de datos del acelerómetro (Noise floor=200uG* $\sqrt{BW}$), se pudo reducir el ruido y aumentar la resolución.

* ANALOG DEVICES, "ADXL202E Low-Cost ± 2 G Dual-Axis Accelerometer with Duty Cycle Output [PDF]",

La alimentación del acelerómetro se realiza a través de uno de los pines de propósito general del microcontrolador el cual proporciona un nivel de tensión de 5V, para este valor de alimentación la sensitividad típica del acelerómetro es de 312mV/G.

La salida nominal de la ADXL202E es: 0g=50% ciclo útil

El factor de escala es: 1g=12.5% cambio ciclo útil

Esto quiere decir que para T=80ms T1=35ms a 1g
T1=40ms a 0g
T1=45ms a -1g

Estos valores son proporcionados por el fabricante, los cuales varían entre dispositivos debido a que no todos quedan exactamente iguales en el momento de su fabricación. Además estos valores son medidos a nivel del mar, por lo cual, los valores obtenidos varían dependiendo de la altura a la que se obtengan.

La señal PWM generada por el acelerómetro es medida en unidades de 2us por el microcontrolador (este muestreo se lleva a cabo por medio de la función PULSIN del software programador del microcontrolador)[*]. Debido a que la señal proporcionada en la salida del acelerómetro es digital, no es necesario un conversor análogo-digital (A/D), lo cual es una ventaja por la reducción de componentes y la disminución de los niveles de ruido de cuantización, además de evitar una posible etapa de amplificación.

[*] Referencia bibliográfica Nº11: PARALLAX INC. "Manual de programación BASIC Stamp 2 Versión 1.1 [PDF]" [http:\\] www.parallaxinc.com

Datos prácticos:

Eje X:

0g ~ 47.6% ciclo útil
Esto quiere decir que para T2 = 68.6ms (34300 unidades de 2us)
T1 ~ 41.1ms a 1g, aumenta 11.47% de ciclo útil (4269 unidades de 2us)
T1 ~ 32.6ms a 0g (16327 unidades de 2us)
T1 ~ 23.8ms a -1g, disminuye 11.9% de ciclo útil (4435 unidades de 2us)

Eje Y:

0g ~ 54.27% ciclo útil
Esto quiere decir que para T2 = 68.6ms (37229 unidades de 2us)
T1 ~ 45.74ms a 1g aumenta 11.32% de ciclo útil (4215 unidades de 2us)
T1 ~ 37.34ms a 0g (18615 unidades de 2us)
T1 ~ 28.76ms a -1g disminuye 11.53% de ciclo útil (4293 unidades de 2us)

De lo que se puede observar que existe un factor de escala de aproximadamente: 11.5% de cambio del ciclo útil por cada 1G, lo que equivale a una variación de aproximadamente 8.5ms en el periodo de la señal.

El acelerómetro fue puesto con sus ejes efectivos paralelos a la superficie, debido a que en esta orientación este es más sensible para tomar mediciones. En esta orientación la sensibilidad a cambios en la inclinación es más alta, el error puede ser bajo y la precisión alta.

Si el acelerómetro esta orientado con el eje de la gravedad (ejes efectivos perpendiculares a la superficie terrestre, es decir, cuando el sensor presenta

aceleración de ±1G lo que equivale a un ángulo de inclinación de 90º), los cambios en la salida por grado de inclinación son insignificantes, ya que el delta de variación por grado de la señal seno se reduce a medida que el ángulo de inclinación se aproxima a 90º (Ver tabla 1) de tal forma que la sensibilidad puede ser baja y el error significativo.

Debido a esto la interpolación utilizada para hallar el valor de inclinación en grados se realizo en intervalos más cortos a medida que el ángulo se aproximaba a 90° (Ver numeral 1.1).

Bajo condiciones normales, la exactitud del inclinómetro es de un grado en el rango de -60 a 60 grados.

1.4 CALIBRACIÓN DEL ACELERÓMETRO

Para la utilización del acelerómetro fue necesario realizar una calibración, debido a que sus condiciones iniciales son determinadas por el fabricante a la altura del nivel del mar donde la gravedad es diferente, para esto fue necesario adquirir mediante una rutina del microcontrolador los valores (en unidades de 2us) para el eje cero y para los valores de gravedad máximo y mínimo. Los valores máximo y mínimo para nuestra aplicación se tomaron de +/-1G debido que, al no existir fenómenos externos que generen aceleración en nuestro dispositivo, la única aceleración a la cual este es sensible es a la gravedad. Esta calibración es necesario hacerla para cada uno de los ejes de forma separada.

El software del microcontrolador *(PBASIC)* tiene una función llamada *DEBUG* que permite ver los valores que entran por los pines E/S de propósito general. Mediante el uso de esta función se tomaron los valores medidos en nuestro eje cero (Es decir a cero grados de inclinación), y en los puntos de gravedad máxima

y mínima para posteriormente ser usados como referencia dentro de las diferentes rutinas de programa **PBASIC** del inclinómetro.

Teóricamente el acelerómetro debe brindar una señal de 80ms, con un ciclo útil del 50% en 0G, lo que equivale a 40ms. Debido a que el microcontrolador toma medidas en unidades de 2us, el ciclo útil es de 20000 unidades. En este caso a pesar de utilizar una resistencia de aproximadamente 10MOhm el periodo es de 68.6ms, esto se debe a que el valor de la resistencia no es exacto. Para la calibración, los datos obtenidos de ciclo útil en 0G fueron de 16327 unidades en el eje X y 18615 unidades en el eje Y.

Para la toma de medidas en los puntos máximo y mínimo (+/-1G), cada uno de los ejes se ubicó de forma perpendicular al plano de referencia, primero hacia abajo para tomar la medida en -1G (Gmin) y posteriormente hacia arriba para la medida de +1G (Gmax). Los valores obtenidos de estas calibraciones fueron:

Ax para Gmax= 20596 unidades
Ax para Gmin= 11892 unidades
Ay para Gmax= 22830 unidades
Ay para Gmin= 14322unidades

Para estas medidas fueron tomadas 10 de cada una de ellas y se obtuvo un promedio, en la tabla 2 se pueden observar las medidas tomadas.

Tabla 2. Medidas para la calibración del acelerómetro.

Medida	X y Y a 0g	X a +1G G-máximo	Y a +1G G-máximo	X a -1G G-mínimo	Y a -1G G-mínimo
X_1	16326	20603	-	11891	-
Y_1	18618	-	22832	-	14322
X_2	16324	20588	-	11894	-
Y_2	18624	-	22831	-	14334
X_3	16329	20603	-	11887	-
Y_3	18615	-	22825	-	14318
X_4	16327	20591	-	11888	-
Y_4	18609	-	22831	-	14322
X_5	16330	20596	-	11878	-
Y_5	18604	-	22830	-	14331
X_6	16329	20599	-	11892	-
Y_6	18613	-	22835	-	14322
X_7	16336	20596	-	11902	-
Y_7	18610	-	22825	-	14333
X_8	16331	20592	-	11887	-
Y_8	18631	-	22831	-	14325
X_9	16324	20593	-	11909	-
Y_9	18617	-	22834	-	14318
X_{10}	16331	20599	-	11901	-
Y_{10}	18606	-	22831	-	14318
$X_{Promedio}$	16327	20596	-	11892	-
$P_{romedio}$	18615	-	22830	-	14322

1.5 SENSITIVIDAD:

Para determinar la sensitividad del acelerómetro se tomaron medidas en los puntos máximo y mínimo (+/- 1G o 90° y -90°) y posteriormente aplicando la formula promocionada por el distribuidor en la hoja de datos del ADXL202E (Anexo 2, página 12), la cual enuncia que:

$$\text{Sensitividad} = [A - B] / 2G \qquad (6)$$

Donde

A = Salida del acelerómetro con el eje orientado a +1G.

B = Salida del acelerómetro con el eje orientado a -1G.

Utilizando un multímetro se midió el voltaje entre cada uno de los condensadores que determina el ancho de banda y la tierra. Estos valores son reemplazados en la formula 6 para determinar la sensitividad en unidades de mV/G, los resultados obtenidos se muestran en la tabla 3.

Con los valores obtenidos en la calibración y aplicando la formula 4, se pudo determinar la sensitividad en unidades de %/G (porcentaje por unidad de gravedad). Un dato importante para la medida de aceleración, es el valor de la sensitividad en unidades de 2us. Este se determina de la misma forma, reemplazando los valores medidos en los puntos máximo y mínimo.

Tabla 3. Sensitividad

EJE	SENSITIVIDAD		
	En unidades de %/G	En unidades de mV/G	En unidades de 2us
X	11.69	375	4352
Y	11.42	370	4254
Valores Típicos*	12.5	312	

En la tabla anterior se observa que el porcentaje de variación obtenido por unidad de gravedad esta dentro del rango típico proporcionado por el fabricante (10.5% a 14.5%)**.

* Estos valores fueron tomados de la hoja de datos del ADXL202E (Anexo 2).

** ANALOG DEVICES, "ADXL202E Low-Cost ± 2 G Dual-Axis Accelerometer with Duty Cycle Output [PDF]", United States Of America 2002

2. INCLINÓMETRO

El inclinómetro es una herramienta de medición de ángulos de inclinación en dos ejes el cual esta conformado por un modulo o panel de control y una pequeña caja donde se aloja el censor, conectados entre si por medio de un cable blindado de cuatro hilos de aproximadamente 50 centímetros de longitud.

El panel de control es una caja plástica de color negro con dimensiones 15cm de ancho x 9cm de profundidad x 5.5cm de alto, la cual consta de un interruptor de encendido (**ON-OFF**), un **switch** selector de dos estados para seleccionar el modo de alimentación (Batería de 9V o fuente DC externa). Para la conexión de la fuente de alimentación externa cuenta con dos conectores (Rojo para el positivo y negro para el negativo de la fuente), esta alimentación puede estar en el rango de +7Vdc a +12Vdc, además cuenta con un puerto serie estándar RS232 DB9 para la conexión con el computador. Un menú compuesto de 4 pulsadores más uno para **Reset** el cual se usa en caso de mal funcionamiento del inclinómetro para reiniciarlo (Ver Tabla 4), y una pantalla de cristal líquido (LCD) para la visualización del menú (Ver Figura 6).

Tabla 4. Menú.

TECLA	FUNCIÓN
1	Confirmación de una medida de inclinación para almacenar en la memoria y posteriormente ser descargada al PC.
2	Ingresar al submenú, para limpiar la memoria o vaciar los datos almacenados. Además de servir como negativa para la limpieza de memoria y/o en la descarga de las medidas de inclinación.

3	Muestra las medidas en los ejes X y Y. Además es usada para confirmar la limpieza de memoria y/o y para la descargar las medidas de inclinación al PC.
4	Salir de cualquier menú o submenú.

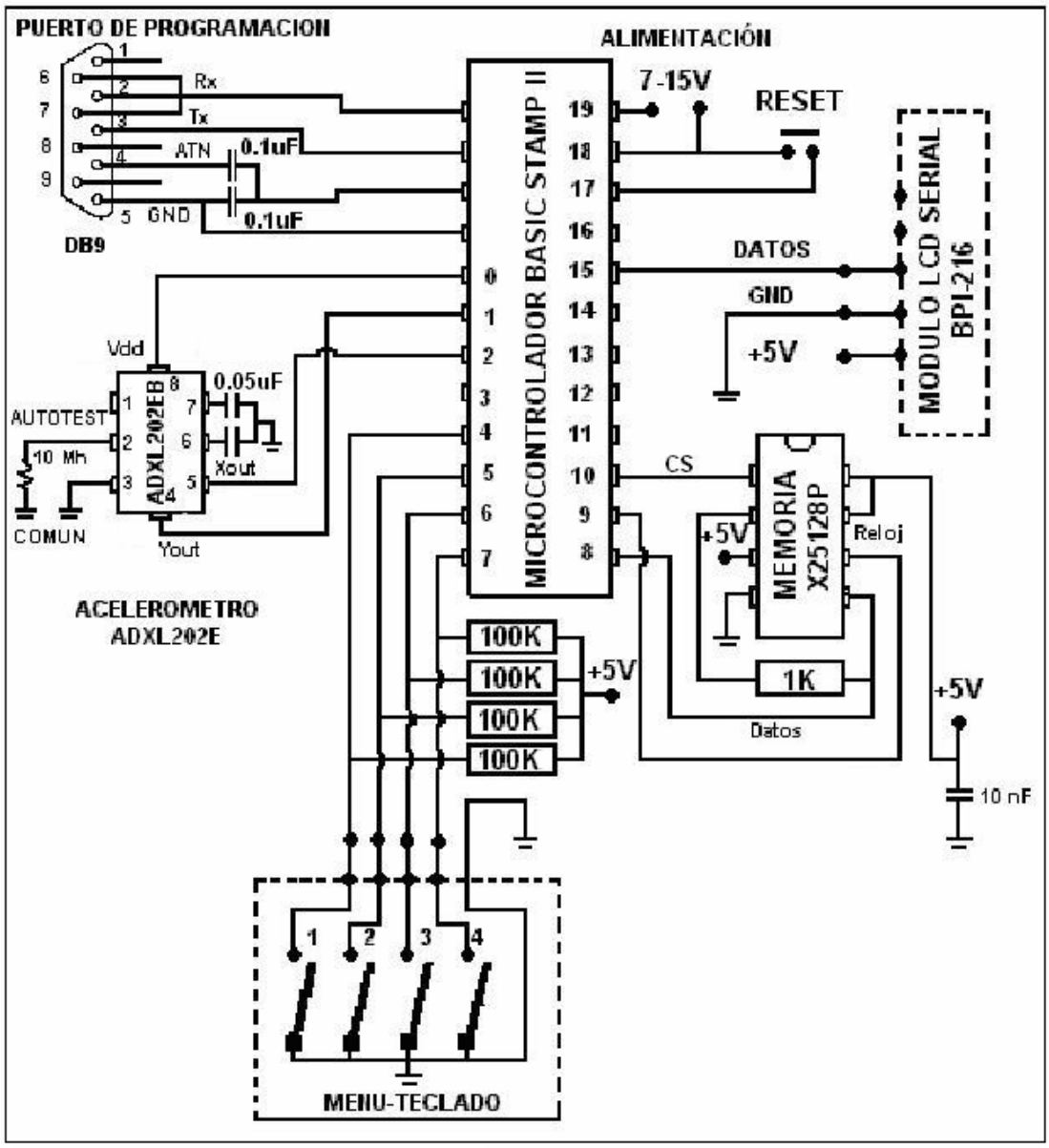
Figura 6. Aspecto físico del inclinómetro

La circuitería se puede dividir en dos etapas, la correspondiente a la etapa de adquisición, cuyo componente principal es el acelerómetro, y la otra etapa que tiene que ver con el procesamiento, almacenamiento, visualización y descarga de datos, en la cual se encuentra el microcontrolador, la memoria externa, el LCD y el puerto serial DB9.

La etapa de alimentación esta compuesta por dos reguladores de voltaje (LM7805 y LM7809) los cuales se encuentran entre los pines de alimentación de la aplicación y las componentes de la segunda etapa, mencionada en el párrafo anterior.

El dispositivo encargado del procesamiento de la señal proveniente del acelerómetro es el microcontrolador, el cual posee 24 pines, 3 de los cuales están conectados al acelerómetro (2 para la adquisición se datos y uno para alimentación), 4 conectados al puerto serial DB9 (Usados para la programación del microcontrolador y descarga de datos al PC), 4 para los pulsadores controladores de los menú y submenús mostrado en el LCD, 3 pines para la conexión con la memoria externa (un pin para el chip select, otro para la transmisión de datos en forma serial y el tercero para el reloj), un pin para la transmisión de datos en forma serial con el LCD, 2 pines para el botón de reset y uno para alimentación. A continuación en la figura 7 se presenta el plano completo del inclinómetro.

Figura 7 . Plano del inclinómetro



La batería utilizada es de +9V, suficiente para aproximadamente 150 horas de operación (este valor se determinó experimentalmente). Para reducir el consumo de potencia del equipo en el modo de alimentación con batería se suprime la utilización del regulador LM7809, ya que no es de utilidad, debido a que la batería no suministra más de 9 voltios, de manera que, solo se utiliza el LM7805 para

regular el voltaje de alimentación del LCD y la memoria externa. Esto conduce a un ahorro en el consumo de potencia. Vale anotar que la eficiencia de los reguladores no es la mejor (aproximadamente del 30%), sin embargo se utilizan debido a su fácil adquisición y por cumplir la función para la que se necesitan.

El cable de comunicación entre inclinómetro y PC tiene características de comunicación serial, además de tener un blindaje para reducir posibles ruidos externos.

Comunicación y descarga de datos. El único software que es necesario para descargar los datos es un programa de comunicación serial. Un simple programa terminal puede ser excelente. Muchos programas terminales están disponibles en software de dominio publico y muy frecuentemente vienen como parte de un sistema operativo (Ej. Hyperterminal).

Los parámetros de comunicación determinan la velocidad y el formato de la comunicación de datos serial. Para que la comunicación del inclinómetro con el computador sea exitosa, sus parámetros de comunicación deben ser los siguientes:

Rata de baudios	9600 bps
Bits de datos	8
Paridad	ninguna
Bit de parada	1
Control de flujo	ninguno

Los computadores están casi universalmente equipados con 1 o 2 puertos seriales estándar RS232 DB9 (COM1, COM2,…). En caso de no tener disponible un puerto DB9, se puede conectar a un DB25 y utilizar un adaptador de DB9 a DB25. Los

pines de conexión requeridos para la comunicación del cable se muestran en la tabla 5.

Tabla 5. Forma de conexión de los pines.

INCLINÓMETRO		-	COMPUTADOR	
PIN	**SEÑAL**	-	**PIN**	**SEÑAL**
5	**GND**	-	**5**	**GND**
3	**TXD**	-	**2**	**RXD**
2	**RXD**	-	**3**	**TXD**

Nota: Los pines 6 y 7 van cortocircuitados para que el PC detecte automáticamente que hay un periférico conectado al puerto.

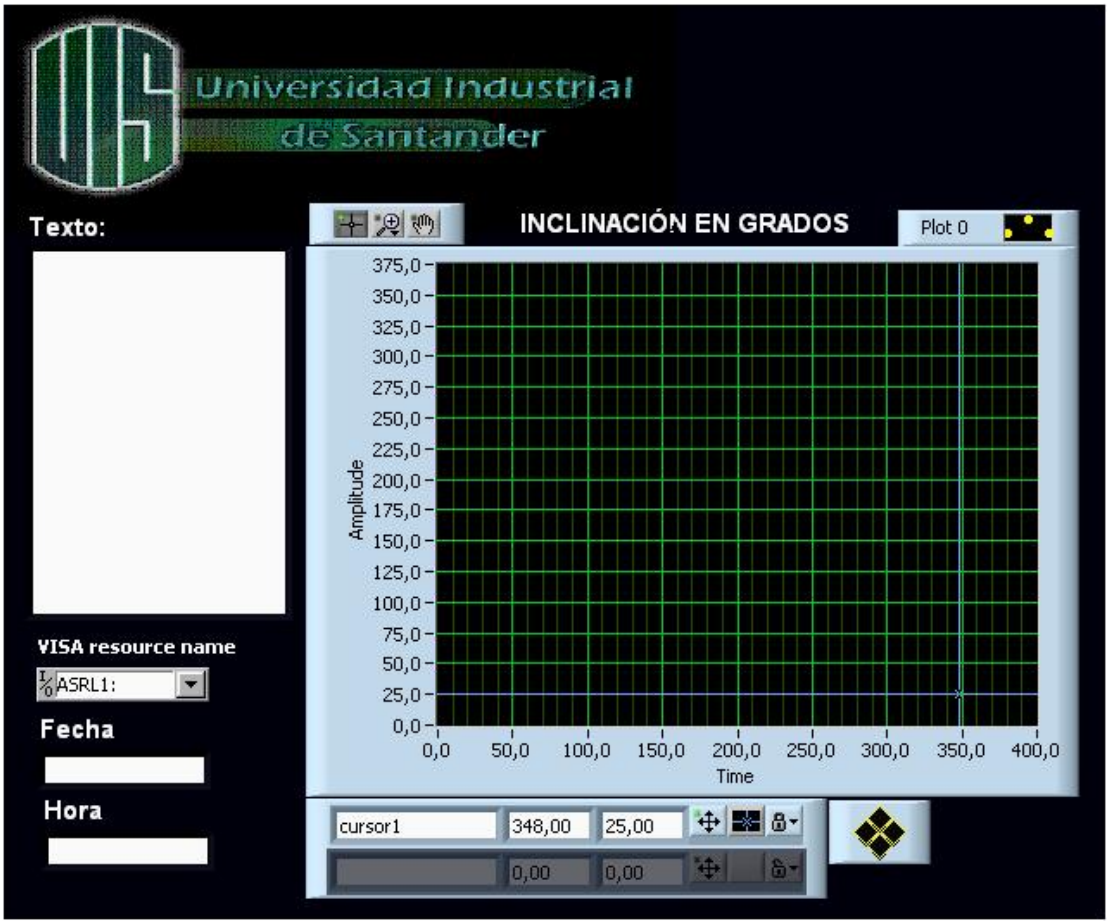
Se debe hacer un cable serial estándar y unir los pines 6 y 7 para auto detección de la conexión al puerto. Esta conexión se hizo basados en el protocolo estándar RS232.

Para la visualización de las medidas de inclinación descargadas al PC, fue diseñado un programa en LabVIEW, este es usado para confirmar la descarga de los datos, visualizando las medidas de inclinación en el PC, además de contar con otra herramienta de visualización de los datos, diferente a la instrucción DEBUG del programa compilador PBASIC.

Esta interfase en LabVIEW (Ver figura 8) muestra dentro de un arreglo los valores correspondientes al número de la muestra, y medida de inclinación (en grados) en cada uno de los ejes. Además son almacenados en un archivo en Excel junto a la fecha y hora (estos parámetro los determina el operario), lo cual se puede utilizar como un historial de medidas de inclinación.

Se debe tener en cuenta que en caso de utilizar LabVIEW se debe contar con la licencia, la cual tiene un costo elevado, sin embargo puede ser reemplazado por cualquier otro programa terminal para comunicación serial (Ej. Hyperterminal en Windows).

Figura 8. Interfase en LabVIEW.



3. HARDWARE

En este capítulo se describen las características que se consideran importantes y que ayudan a un mejor entendimiento del diseño y funcionamiento del inclinómetro.

3.1 MICROCONTROLADOR BASIC STAMP

Los BASIC Stamps son microcontroladores híbridos de 8 Bits diseñados y desarrollados por Parallax inc. Su nombre se debe, Basic al lenguaje de programación que usa y Stamp a que su tamaño es similar al de una estampilla. Estos son usados en una gran cantidad de aplicaciones. Muchos proyectos que requieren incluir un sistema con varios niveles de inteligencia pueden usar un modulo Basic Stamp como controlador.

Cada Basic Stamp usa un chip intérprete, las primeras usan un PIC de Microchip Technology Inc. (hasta la Basic Stamp 2) y las demás un chip SCENIX. Cada Basic Stamp viene con un chip interprete BASIC, una memoria interna (RAM y EEPROM), un regulador de 5V, un número de pines E/S de propósito general (TTL, 0-5 V), y un *set* internamente construido de comandos para operaciones matemáticas y operaciones entre pines E/S.

Los Basic Stamps son capaces de ejecutar miles de instrucciones por segundo y son programados con un simple y personalizado lenguaje de programación BASIC, llamado PBASIC. Este programa es almacenado dentro de una EEPROM serial en la board del Basic Stamp, la EEPROM es una memoria no volátil, esta retiene memoria incluso sin alimentación. La memoria EEPROM usada en los Basic Stamps es garantizada por 40 años y por 10.000.000 ciclos de memoria escritos.

Todos los Basic Stamps se alimentan con 6V a 15V, utilizando su regulador interno de 5V para rectificar una entrada de 6V a 15V (en el pin VIN) a los 5V que requieren los componentes internos de los Basic Stamps. Si se alimenta con una fuente entre 6V a 15V, esta se debe conectar directamente a los pines VIN y GND del Basic, esto es valido para todos los modelos del Basic, pero si se alimenta con una fuente regulada de 5V, se debe conectar directamente a los pines +5V y GND.

Los pines de E/S de los Basic Stamps pueden ser usados como entradas y salidas con TTL/CMOS con niveles de señal de 0 - 5V. Sin embargo se puede utilizar algunos comandos y técnicas especiales para los límites de entrada y salida de señales análogas. Los comandos RCTIME y PWM pueden ser usados para leer una resistencia variable o una salida de voltaje variable de 0 a 5V. Los pines de E/S no pueden leer por ellos mismos voltajes análogos, sin embargo pueden hacerlo por medio de una interfase con un conversor A/D.

Los Basic Stamps trabajan en rangos de temperatura entre 0º C y 70º C, si no se trabaja dentro de este rango, se recomienda usar los Basic Stamps industriales, los cuales trabajan entre -40 ºC y 125 ºC (para mayor información visitar www.parallaxinc.com).

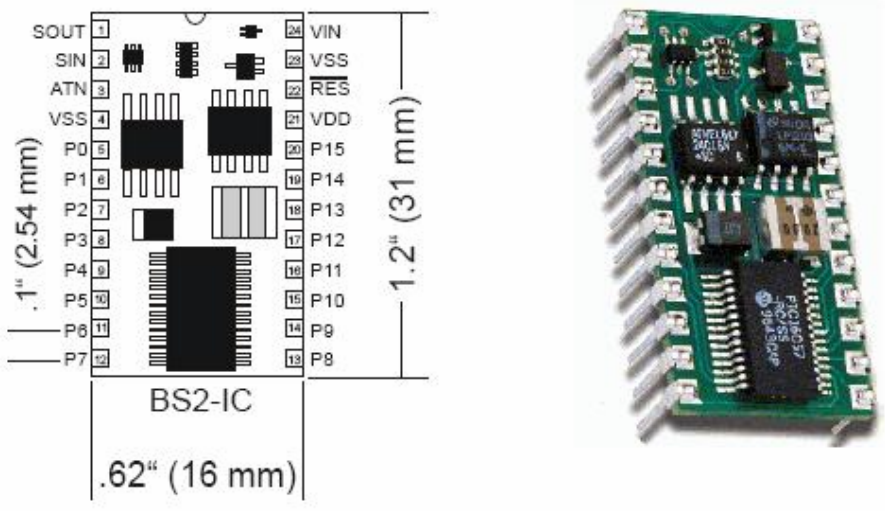
La impedancia de entrada de los pines de los Basic Stamps es aproximadamente de $1M\Omega$.

3.1.1 Clases de Basic Stamps. Actualmente existen seis modelos funcionales del Basic Stamp; el Basic Stamp 1, Basic Stamp 2, Basic Stamp 2e, Basic Stamp 2sx, Basic Stamp 2p y Basic Stamp 2pe; y 9 versiones físicas. A continuación se mostrarán algunas características del Basic Stamp utilizado en la construcción del inclinómetro.

3.1.2 Basic Stamp 2. El Basic Stamp 2 está disponible en dos diferentes tipos de paquetes físicos. El BS2-IC (Figura 9) usa componentes de montaje superficial para montar en un pequeño paquete DIP de 24 pines y el OEMBS2 se caracteriza como un esquema fácil por sus pistas para clientes que desean integrar el Basic Stamp 2 directamente en un solo circuito para una reducción de componentes. Las características funcionales de este modelo son:

- Tiene 16 pines de E/S de propósito general, los cuales manejan entre 40mA y 50mA por grupos de 8 pines y 2 para conexión serie.
- Ejecuta aproximadamente 4000 instrucciones por segundo.
- Requiere una interfase serial para su programación.
- Espacio en memoria entre 500 y 600 líneas de código, repartida en 2K octetos de memoria de programa y 32 octetos de memoria RAM (6 para las E/S'^{s} y 26 para las variables de programa).
- Consume 8mA en estado activo y 100uA en estado inactivo, sin incluir circuitos en los pines de E/S.

Figura 9. Basic Stamp 2 (BS2-IC).



Fuente: PARALLAX INC. "BASIC Stamp Programming Manual 2.0c"
PARALLAX INC. "Manual de programación BASIC Stamp 2 Versión 1.1"

Tabla 6. Descripción de pines del Basic Stamp 2.

PIN	NOMBRE	DESCRIPCIÓN
1	SOUT	Salida serial: Se conecta al pin RX (pin 2 en DB9 / pin 3 en BD25) del puerto serial del PC para programación.
2	SIN	Entrada serial: Se conecta al pin TX (pin 3 en DB9 / pin 2 en BD25) del puerto serial del PC para programación.
3	ATN	Atención: Se conecta al pin DTR (pin 4 en DB9 / pin 20 en BD25) del puerto serial del PC para programación.
4	VSS	La tierra del sistema:(el mismo pin 23) Se conecta al pin GND (pin 5 en DB9 / pin 9 en BD25) del puerto serial del PC para programación.
5-20	P0-P15	Pines de E/S de propósito general, cada uno puede manejar entre 20 mA y 25 mA, sin embargo, el total de todos los pines no podrá exceder los 40 mA a 50mA fuente usando el regulador interno de 5V. El total por grupos de 8 pines (P0 – P7 ó P8 – P15) no podrá exceder los 40mA a 50 mA fuente usando el regulador interno de 5V.
21	VDD	Entrada / Salida 5V DC: Si un voltaje irregulado es aplicado al pin VIN, este pone 5V en su salida. Si no se aplica un voltaje al pin VIN, entonces un voltaje regulado entre 4. 5V y 5.5V debe ser aplicado a este pin.
		Reset entradas y salidas, se pone en bajo cuando la

22	RES	alimentación esta por debajo de los 4.2 V aproximadamente. Puede ser manejada en bajo forzando a un reset. Este pin puede ser manejado internamente y desconectarse si no se necesita. No lo maneje en alto.
23	VSS	La tierra del sistema:(el mismo pin 4) se conecta al terminal tierra (GND).
24	VIN	Alimentación irregulada: Acepta entre 5.5 – 15 VDC (6-40 VDC en la BS2-IC rev. e), qué se regula internamente a 5 voltios. Puede desconectarse si se aplican 5 voltios al pin VDD (+5V).

Fuente: BASIC Stamp Programming Manual 2.0c.

3.1.3 Lenguaje PBasic. El lenguaje PBASIC fue creado específicamente para los Basic Stamps, este es un lenguaje simple y fácil de aprender. Este incluye muchas de las instrucciones caracterizadas en otras formas de lenguaje Basic (GOTO, FOR…NEXT, IF…THEN), así como algunas instrucciones especializadas (SERIN, PWM, BUTTON, OUNT y DTMFOUT). Existen diferentes versiones dependiendo del modelo del Basic con el que se esté trabajando, en esta aplicación se utiliza la versión Stampw.exe (v. 2.0).

Las siguientes son las especificaciones mínimas requeridas para usar el software editor del Basic Stamp:

- Computador compatible IBM 80486 (80286 para DOS) (o superior).
- Sistema operativo Windows 95/98/NT4/2000 (DOS 5.0 o para versiones superiores de DOS).
- 16 Mb de RAM (1 Mb para DOS).
- 1 Mb de espacio disponible en el disco duro.
- Drivers CD-ROM.
- 1 puerto serial estándar RS232 disponible.

3.1.4 Conexión para la programación del Basic Stamp II. Se debe tener cuidado pues muchos de los problemas al programar el BASIC STAMP son el resultado de un cable mal hecho o de mala conexión al programarlo en el circuito.

Es vital que se verifiquen sus conexiones y verifique los pines para evitar problemas de comunicación con el Basic Stamp.

Una vez instalado el software del Basic Stamp (este software es proporcionado por Parallax de forma gratuita en su pagina Web www.parallaxinc.com), conecte el Basic Stamp al puerto serial.

La programación del Basic Stamp II se hace a través del puerto serial también conocido como DB9 o DB25 (la diferencia radica en el numero de pines del conector), este al igual que el puerto paralelo se encuentra en la parte trasera de una computadora personal.

En la parte superior de la ventana principal del PBASIC se tiene la opción de elegir el tipo de Basic con que se va programar, después de haber conectado correctamente todos los circuitos incluyendo el puerto de comunicación (serial o paralelo), ya se puede a ejecutar el software editor del Basic Stamp.

3.1.5 Pautas y precauciones. Cuando se usa el Basic Stamp se deben tener en cuenta las siguientes precauciones:

- Esté alerta a la sensibilidad estática de los dispositivos y a las situaciones estáticas. El BASIC STAMP, como otros circuitos integrados (IC's), pueden ser afectados por descargas estáticas que normalmente ocurren cuando tocamos superficies conectadas con tierra u otros conductores. Las condiciones ambientales (cambios de humedad, viento, superficies estáticas, etc.) juegan un papel importante en la presencia de cambios de cargas estáticas. Siempre

es recomendable usar correas antiestáticas y conectadas con tierra o una esfera disipadora estática cuando trabajamos con dispositivos como el BASIC Stamp, además de mantenerlos boca abajo cuando se estén manipulando.

- Verifique que toda alimentación este apagada antes conectar/desconectar. Si el BASIC Stamp está conectado a la alimentación o algún dispositivo está conectado mientras se inserta o remueve de un circuito, el BASIC o el circuito podrían resultar dañados.

- Verifique la orientación del BASIC STAMP antes de ser conectada a su circuito y otros dispositivos. Como otros circuitos integrados, el BASIC STAMP debe ser insertado en una orientación específica en relación a su circuito. Alimentando el circuito con un circuito integrado conectado al revés causa daños al circuito integrado y a otros componentes en el circuito. Los módulos de 24 pines (BS2, BS2e, etc.) tienen un indicador de medio círculo en la parte superior del módulo. Este indica (al coincidir el modulo con el medio circulo arriba) que el pin numero 1 es el primer pin en la parte superior izquierda del dispositivo. La conexión que acepta este módulo de 24 pines también tiene un medio círculo o muesca en un extremo, indicando la orientación correcta.

- No se deben conectar los pines de salida directamente a una fuente externa de poder o correrá el riesgo de destruir el Basic Stamp.

- Trabajar siempre entre los rangos de temperatura correspondientes a cada Basic Stamp (0º C a 70º C), si la aplicación requiere rangos de temperatura mayores, se recomienda usar las Basic Stamps industriales, las cuales trabajan entre rangos de temperatura mayores (-40º C a 125º C).

3.2 LCD BPI216 de Scott Edwards Electronics Inc., USA. Para la elección de la LCD se tuvo en cuenta la recomendación del fabricante del Basic Stamp, la cual

está fundamentada en el hecho que se hace un ahorro en líneas de código (PBASIC) y que el envío de datos hacia el BPI216 se realiza a través de 1 solo pin E/S de propósito general.

3.3 Memoria X25128P de XICOR. Por recomendación del fabricante del Basic Stamp, se selecciona esta memoria. Lo cual se fundamenta en el hecho que se hace un ahorro en líneas de código al momento de una función de escritura y lectura. Otro punto importante es que se puede aumentar la capacidad de almacenamiento en caso de ser necesario, lo cual se realiza adicionando memorias conectadas por el mismo bus y controladas por el pin CS (Chip Select) de la memoria. Para seleccionar en cual de las memorias se desea escribir o leer, se le debe poner el CS en bajo y los de las otras memorias en alto.

3.4 Acelerómetro ADXL202E

Sobre este dispositivo se hace referencia en el capítulo 1.

3.5 Planos de los circuitos impresos

Existen efectos que son dañinos para el desarrollo óptimo de los circuitos tales como resistencias de fuga, caídas de voltaje en laminas de tierra entre otros; en adición, la tendencia de los PCB's (Printed circuit boards) a absorber la humedad atmosférica. Cambio en la humedad es una de las causas que ayudan a la contribución de algunos efectos parásitos.

En las tarjetas de circuitos impresos (PCB's) diseñados tanto para la etapa de adquisición, como para la etapa de procesamiento, almacenamiento y visualización de datos, se siguieron algunas de las siguientes pautas :

- "Cada pulgada cuadrada de la tarjeta de circuito impreso que no contenga circuitos o caminos conductores debe ser un plano de tierra. Violar esa simple regla, invita al desastre. Pero algunas veces, la adherencia estricta a la regla no es garantía de éxito, si la densidad del circuito es muy alta."[*]
- "Si un conductor vulnerable es rodeado por un anillo de guarda, los efectos de las resistencias de fuga serán minimizados."[*]
- "Si los nodos de alta impedancia son rodeados por un anillo conductor, el cual debe estar muy cerca de los nodos, la corriente de fuga será minimizada."[*]
- "Pistas más cortas hacen al circuito menos sensible a interferencias radiadas. Por ejemplo, una longitud de pista de 6 cm. corresponde con la longitud de onda de una señal de 5 GHz por lo que actuaría como una antena perfecta a esa frecuencia y sus múltiplos. Reduciendo la pista a longitudes máximas de 1'8 cm. reducimos la longitud de onda a señales de 16'6 GHz y sus múltiplos."[**]
- "Además es conveniente que los cambios de sentido de las pistas se realicen con ángulos de 45º".[**]

A continuación se muestran los planos utilizados en la construcción del inclinómetro[***]. En la Figura 10 se muestra el diseño del plano donde se aloja el acelerómetro, los condensadores de 0.47uf utilizados para determinar el ancho de banda y la resistencia de 1MOhm que determina el periodo de la señal de salida del acelerómetro. En la figura 11 se muestra el circuito impreso correspondiente a la etapa de procesamiento, almacenamiento, descarga y visualización de datos.

---

[*] ANALOG DEVICES, APPLICATION REFERENCE MANUAL, APPLICATION Analog Devices Inc. 1993.

[**] Ignacio Moreno Velasco, Instrumentación electrónica "Apuntes",Universidad Politécnica de Catalunya, Área de Tecnología Electrónica

[***] A pesar que las recomendaciones anteriores son para PCB's de dos superficies y que trabajan a alta frecuencia, fueron tenidas en cuenta.

Figura 10. Circuito impreso de la etapa de censado

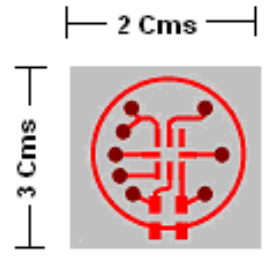
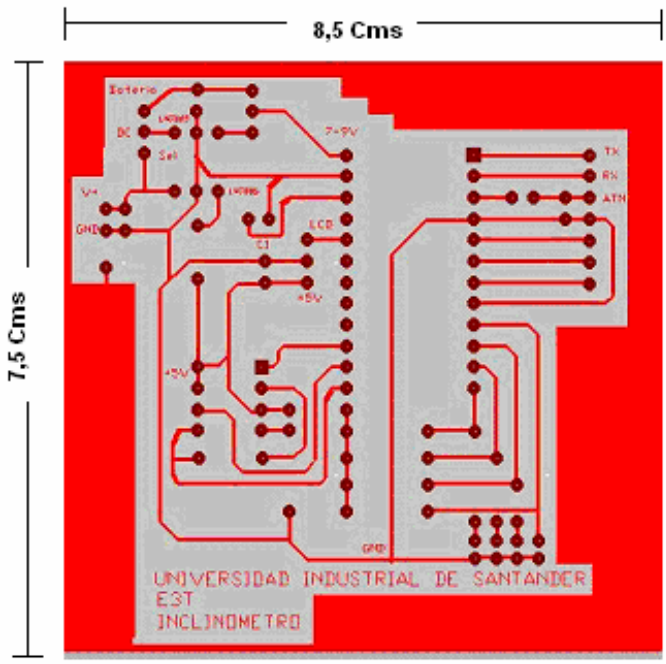


Figura 11. Circuito impreso principal



4. SOFTWARE DEL INCLINÓMETRO

Después de haber interpretado las características de cada uno de los dispositivos usados para el hardware del inclinómetro, se procedió a la programación del microcontrolador. Para esto se descargó el programa compilador de la página del distribuidor*, el cual es gratuito. Para la comunicación entre el PC y el microcontrolador fue construido un cable para puerto serial, por medio del cual se descarga el programa al microcontrolador, y se almacena en la memoria interna de solo lectura programable y borrable eléctricamente.

Para la programación se tuvieron en cuenta los manuales de programación y la ayuda que proporciona el compilador (PBASIC2).

4.1 DESCRIPCIÓN.

Debido a lo extenso del programa en PBASIC, se decidió no incluirlo en este documento, sin embargo con la ayuda del diagrama de flujo y las demás aclaraciones pertinentes se ayuda a la comprensión de las diferentes etapas de programación.

Para el programa del inclinómetro se usan 18 de las instrucciones que conforman el set del **PBASIC**. Se busca la mayor eficiencia del programa dentro de las limitaciones del manejo de programación en **PBASIC.** En la tabla 7 son detalladas cada una de las instrucciones utilizadas junto a una breve descripción de la función que cumplen dentro del código.

* Ver www.parallaxinc.com

Tabla 7. Instrucciones utilizadas para el software del inclinómetro

INSTRUCCIÓN	FUNCIÓN DENTRO DEL CODIGO
HIGH	Poner los diferentes pines E/S en modo alto. Esta instrucción fue usada para alimentar el acelerómetro desde el microcontrolador, y para detener la escritura en la memoria externa.
LOW	Poner los diferentes pines e/s en modo bajo. Deshabilitar el auto-test del acelerómetro, y para iniciar la escritura en la memoria externa.
PAUSE	Hacer una pausa en unidades de milisegundos. Se utilizó después de realizar alguna definición u otra función que requiera un tiempo de espera mayor a un ciclo de maquina para ser ejecutada.
INPUT	Activar un pin de E/S como entrada. Con esta instrucción se definieron como entrada los pines conectados a los ejes del acelerómetro (Canal X y Canal Y).
READ	Lee una variable de tamaño Byte almacenada en la memoria del microcontrolador. Se usó para leer las variables almacenadas en la memoria y saber el contenido de la misma.
GOSUB	Llama una subrutina de programa señalada por una etiqueta. Esta rutina es una de las más usadas en el programa, debido a que todas las operaciones se realizaron en subrutinas separadas (Ej. Escritura en memoria, leer estado de los pulsadores, conversión de datos, etc.).
SEROUT	Salida serial por los pines del microcontrolador. Por medio de esta instrucción se pudo escribir en la LCD, para poder realizar el menú.
NAP	Espera corta para ahorro de energía. Después de cada mensaje enviado al LCD, se efectúa una espera por medio de esta función, esta instrucción es útil principalmente cuando se está alimentando con la batería debido a que disminuye el consumo de potencia del inclinómetro y de esta forma aumenta su durabilidad.
BUTTON	Lee el estado de los pulsadores. Gracias al uso de esta instrucción se puede leer el estado de cada uno de los pulsadores, con los cuales se puede tener acceso a los diferentes submenús y tomar decisiones con respecto a la descarga, almacenamiento, toma de medidas, etc.

GOTO	Salta a una posición especifica del programa. Esta instrucción es usada después de leer el estado de los pulsadores de forma repetitiva hasta que alguno se presione.
PULSIN	Mide el ancho del pulso con un muestreo de 2us, mediante el uso de esta instrucción se lee el ancho del pulso en cada uno de los ejes, para posteriormente ser procesado y determinar la medida de inclinación.
IF…THEN	Instrucción de condicionalidad. Con esta instrucción se hacen las diferentes comparaciones para evaluar si los valores medidos están dentro del rango de medida.
WRITE	Graba un Byte en la memoria interna del microcontrolador. Se usa para asignar valores de tamaño Byte a una variable en la memoria interna.
RETURN	Retorna desde una subrutina a la siguiente línea desde donde fue invocada la subrutina. Esta instrucción fue implementada en cada una de las subrutinas del programa.
FOR…NEXT	Forma un LOOP repetidor entre FOR y NEXT. Para la rutina de borrado de la memoria, en la cual se hace un barrido para borrar datos almacenados en cualquier posición de memoria.
SHIFTOUT	Desplazamiento de datos de salida hacia un dispositivo periférico serial síncrono. Esta instrucción es usada al enviar datos para que sean almacenados en la memoria externa.
SHIFTIN	Desplazamiento de datos de entrada desde un dispositivo periférico serial síncrono. Se usa para revisar si hay espacio en la memoria externa o si está llena en su totalidad y proceder a descargar los datos al PC
DEBUG	Esta instrucción permite ver los datos almacenados en una variable del programa, con esta se puede hacer la lectura de las medidas de obtenidas por el acelerómetro en cada uno de sus ejes (en unidades de 2us).

4.2 PROGRAMACIÓN EN DIAGRAMA DE BLOQUES.

Para la comprensión del programa del inclinómetro es necesario explicar el funcionamiento de las diferentes subrutinas por medio de diagramas de flujo. En las siguientes figuras se pueden observar los diagramas de bloques para las diferentes subrutinas del programa.

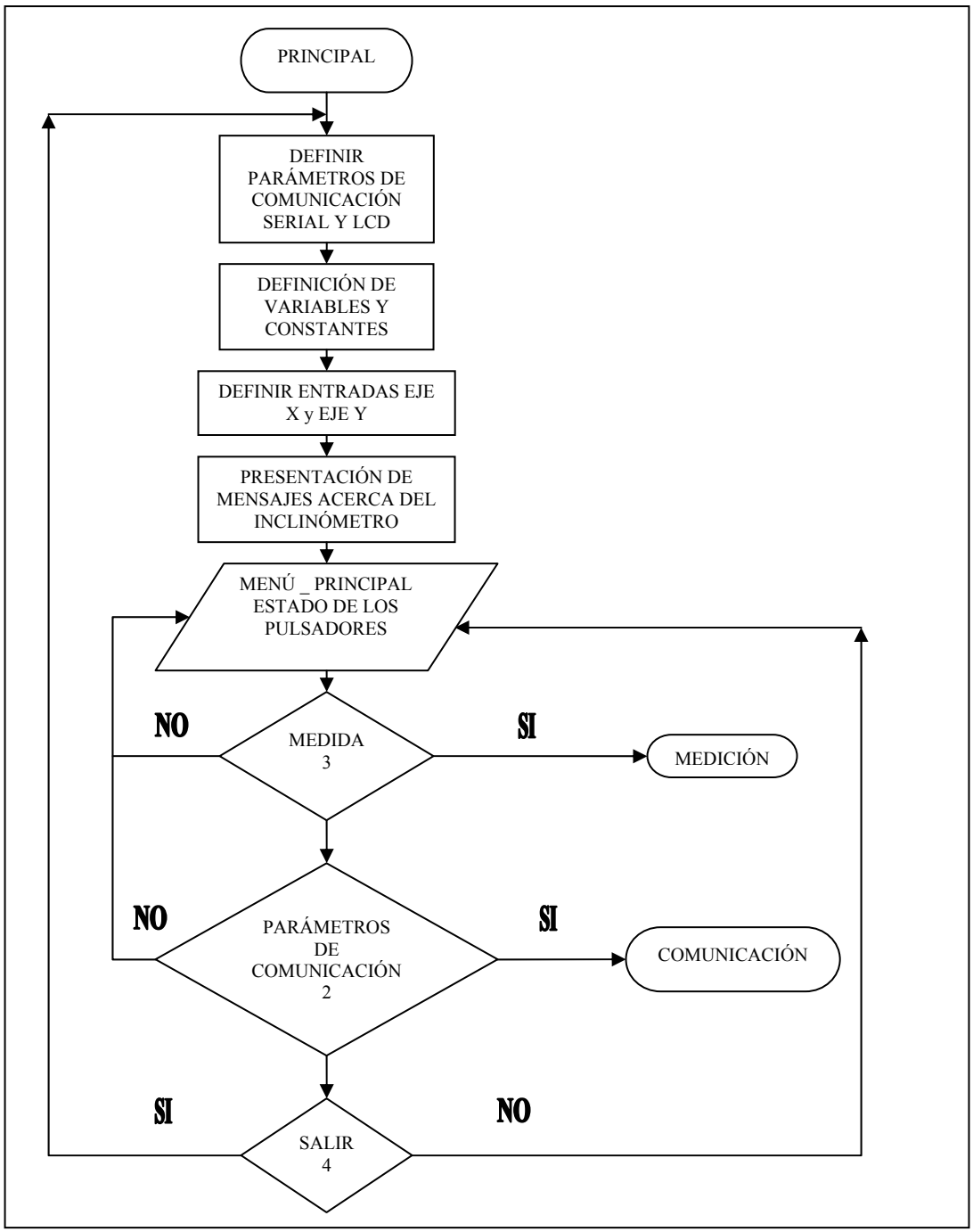
Por medio de estos diagramas se busca que el lector comprenda el funcionamiento del programa, sin necesidad de tener conocimientos en lenguaje PBASIC.

Para evitar confusiones no fueron explicadas cómo se realizan las diferentes operaciones, los diagramas se enfocan hacia como se realiza el desplazamiento dentro de los diferentes menús del código con los mismos nombres con que se identifican dentro del programa. Por lo tanto si se desea profundizar en la forma en que se realizaron las operaciones se recomienda detallar el código, el cual se adjunta a este documento.

La determinación del ángulo de inclinación a partir de la aceleración se detalla en el numeral 1.1 ‘’DETERMINACIÓN DE LA INCLINACIÓN APARTIR DE LA ACELERACIÓN’’.

A continuación se presentan estos diagramas:

Figura 12. Diagrama de bloques general

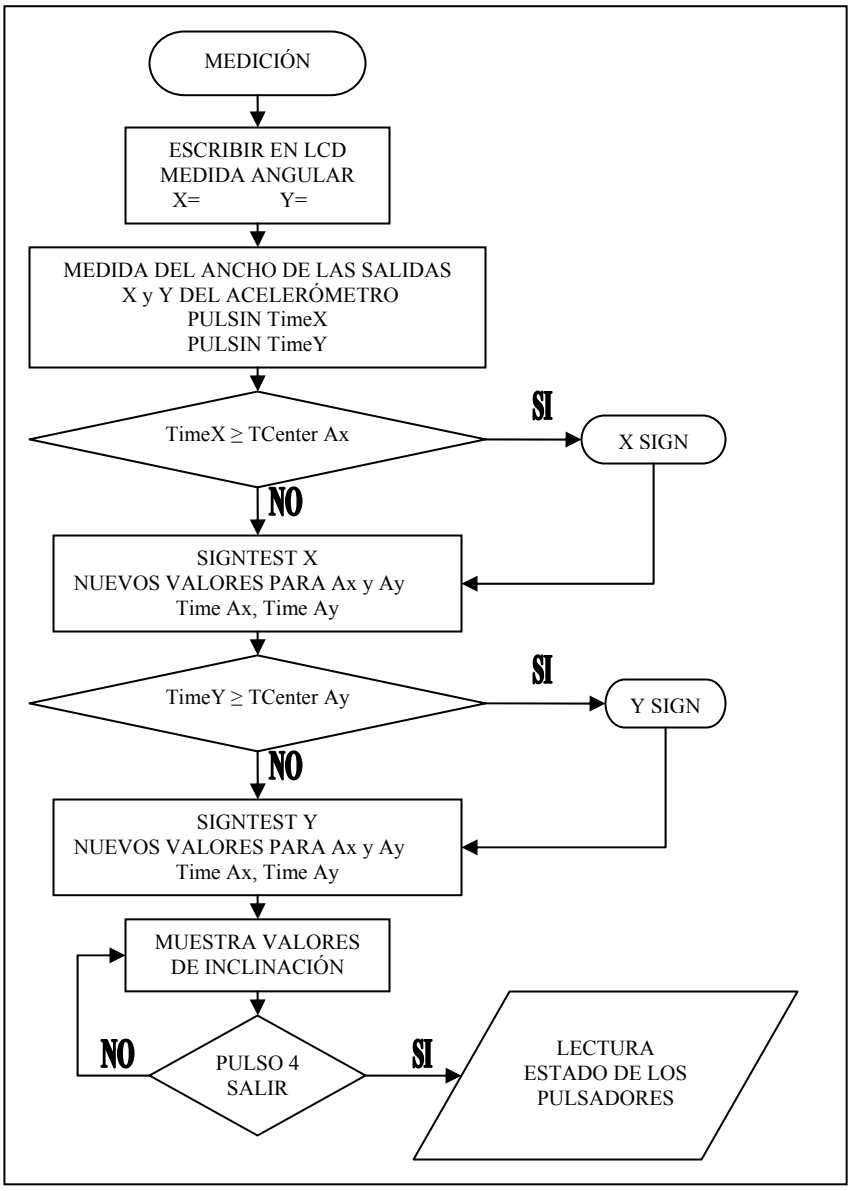


Lo primero que se hace al escribir un código en PBASIC, es definir las diferentes variables y constantes que serán usadas a lo largo del programa[*]. En el código del inclinómetro luego de definir constantes y variables, se procedió a definir como

[*] Ver Referencias [10] y [11].

entradas a los pines conectados a cada uno de los dos ejes del acelerómetro. En pantalla se muestra el menú en donde se puede seleccionar si se quieren tomar medidas [3 MEDIDA] o ingresar a las funciones de comunicación [2 PARAMETROS DE COMUNICACIÓN].

Figura 13. Subrutina para realizar las medidas de inclinación



En esta subrutina se realizan los diferentes cálculos para hallar la medida de inclinación en grados, los cuales son mostrados en pantalla. En las subrutinas XSIGN y YSIGN simplemente se pone en 1 la variable en donde se almacenan los valores temporalmente.

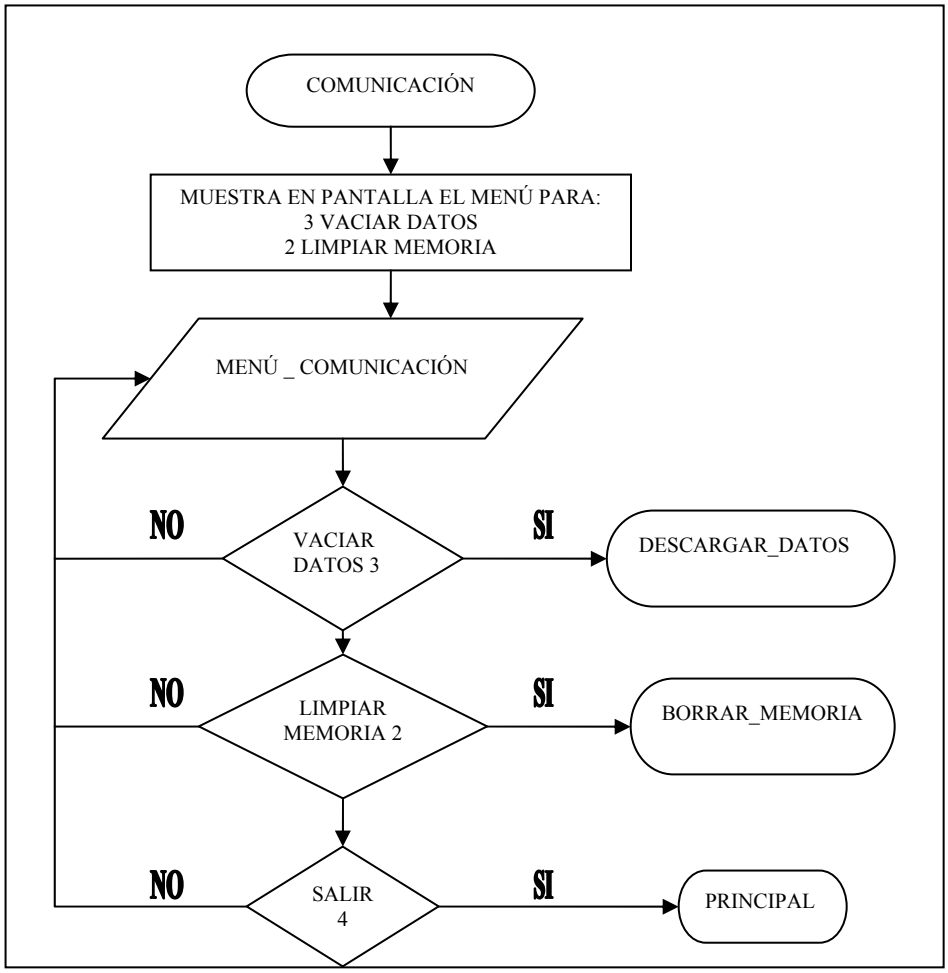
Si cuando en pantalla se muestran las mediciones de X y Y se pulsa 1, se pide la confirmación del dato para ser almacenado, en caso de confirmarlo (pulsando nuevamente 1): se activa el pin CS (Chip Select) en bajo, ejecuta la instrucción SHIFTOUT para enviar los datos a la memoria, de esta forma son guardados los datos. Cuando se termina la transmisión el pin CS es puesto nuevamente el alto.

En el capitulo 2 se da una explicación de las formulas utilizadas para la determinación de la aceleración a partir de las medidas tomadas de cada uno de los ejes del acelerómetro en unidades de 2us. Esta medida se realiza utilizando la función **PULSIN**[*] para el eje X y eje Y.

Por medio de la función **PULSIN** se realiza un muestreo de 2us, el cual arroja un valor (en unidades de 2us) que es evaluado en la formula 5 donde se determina el ciclo útil restando del valor leído (TimeX) el asignado en la calibración al valor de aceleración en 0G (TCenterX) y posteriormente se dividió por la sensitividad[**]. Después de tener este valor de aceleración se realiza la interpolación, donde se determina el ángulo de inclinación.

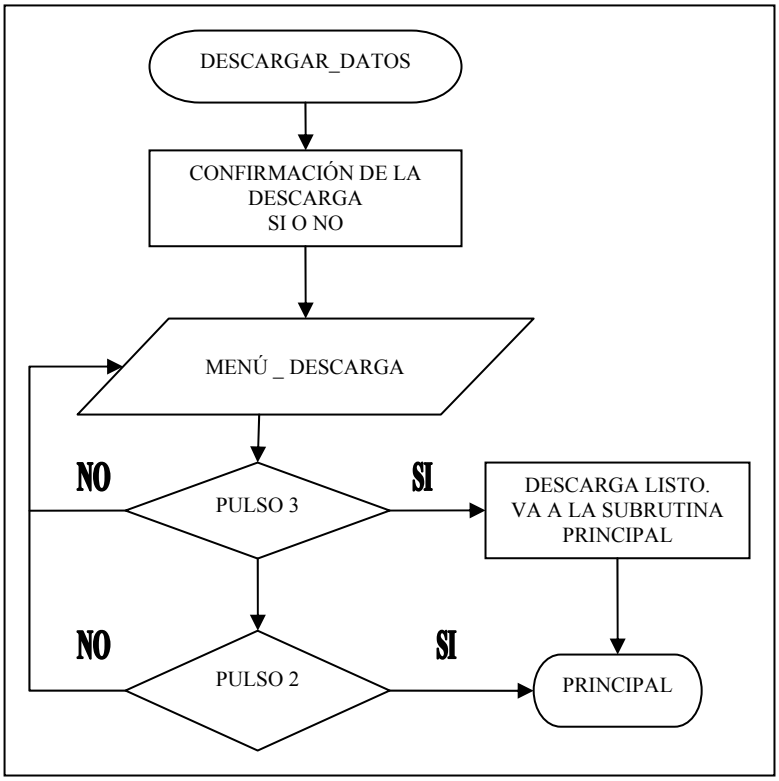
[*] Ver Referencias [10] y [11].
[**] Ver Capitulo 1

Figura 14. Subrutina para seleccionar si descarga datos a PC o borra memoria.



En esta subrutina se despliega en pantalla la pregunta si desea vaciar los datos almacenados en la memoria o si desea limpiarla. Por medio de la función **BUTTON** se evalúa el estado de los pulsadores para invocar la subrutina seleccionada según la decisión tomada. En caso de pulsar 4 se sale al menú anterior.

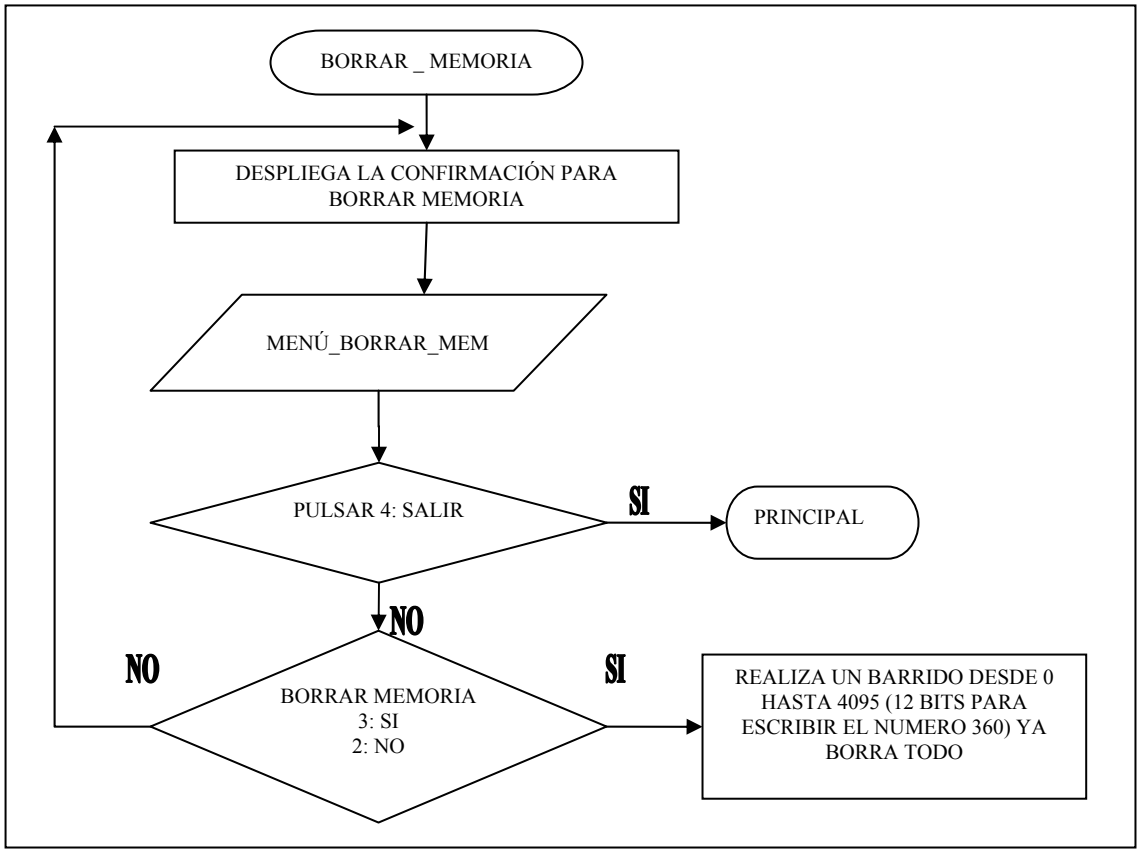
Figura 15. Subrutina para descarga datos a PC.



En esta subrutina se pide la confirmación para la descarga de datos, en caso de ser confirmada, con la instrucción **SEROUT** son descargados la totalidad de los datos almacenados. Para esto se realiza un barrido por medio de la instrucción FOR…NEXT a través la memoria.

La descarga se realiza de forma serial a través del puerto del inclinómetro hacia el PC donde por medio de una interfase en LabView pueden ser visualizados los datos. Para salir de este menú en cualquier momento basta con pulsar 4.

Figura 16. Subrutina para borrar memoria.



Para realizar el borrado de la memoria se realiza un barrido por medio de la instrucción FOR…TO a todas las posiciones de memoria. Este barrido es realizado desde 0 hasta 4095, debido a que se utilizan 12 Bits para escribir un número en memoria (000000000000-111111111111). Se utilizaron 12 Bits (1 + ½ Byte) para poder expresar el número 360, para el cual es necesario expresarlo mínimo en 9 Bits.

5. RESULTADOS OBTENIDOS

El consumo de potencia del inclinómetro fue medido cuando se alimenta con la batería, la cual en el momento de tomar las medidas proporcionaba un voltaje de aproximadamente 7.6 [V]. Los valores de consumo de potencia se muestran en la tabla 8.

Tabla 8. Consumo de potencia del inclinómetro con batería

FUNCIÓN	CONSUMO DE CORRIENTE [mA]	VOLTAJE	POTENCIA [mW]
Medición	10.3	7.66	78.9
Almacenamiento en memoria	10.5	7.66	80.43
Oprimir pulsadores	7.6	7.66	58.21
Descarga de datos	7.4	7.66	56.68
Vaciar memoria	11.1	7.66	80.03

En la tabla anterior se ilustran los consumos de potencia para las diferentes funciones del inclinómetro. A pesar de no haber utilizado las características de baja potencia de los diferentes dispositivos, el consumo de potencia no es tan elevado.

Para disminuir el consumo de potencia se aisló el regulador LM7809 cuando la alimentación está siendo proporcionada por la batería, además se utilizó dentro del código de programación la función **NAP** para poner en modo de baja potencia al microcontrolador en instantes donde no fuera necesario su funcionamiento, como por ejemplo cuando se realiza la visualización de mensajes en el LCD.

Para realizar las pruebas finales de comprobación de ángulos de inclinación se tomo una base móvil con referencia a 0 grados sobre la cual se apoyo el acelerómetro y una base con medidas de ángulos (a escala de un grado). La base móvil se ajustaba a cada medida de inclinación para verificar los valores visualizados en la pantalla del panel de control. Los resultados obtenidos de estas pruebas arrojaban un margen de error máximo de un grado, pero a medida que el ángulo de inclinación se acercaba a 90 grados el error aumentaba entre 1 y 3 grados. Estas pruebas fueron realizadas para cada uno de los ejes del acelerómetro.

Algo importante que se debe tener en cuenta es la inclinación de la superficie en donde se calibró el inclinómetro, debido a que no se dispone de una herramienta con buena exactitud que garantice la ausencia de inclinación de esta superficie, por lo tanto las medidas que se tomen siempre serán con respecto a la superficie donde fue calibrado. En caso que se necesite cambiar este plano de referencia, se debe realizar una nueva calibración para determinar los nuevos valores para las constantes que corresponden a la medida del ciclo útil de la señal del acelerómetro (en unidades de 2us) para los puntos de inclinación cero, máximos y mínimos de cada eje.

Como prueba final se descargaron por el puerto serial del inclinómetro al PC las medidas de inclinación almacenadas en la memoria externa, y se visualizaron por medio del programa diseñado en lenguaje LabVIEW, graficadas estas medidas y almacenadas en un archivo Inclinación.xls. Donde se pueden diferenciar claramente el número de la medida y su valor para cada uno de los ejes.

Inicialmente se tenía un ancho de banda de 100Hz determinado por un valor de los condensadores de 0.05uF, lo cual repercutía en la resolución y estabilidad de la medida de inclinación debido a la cantidad de ruido que producía tal ancho de banda.

$$\text{NOISE FLOOR} = 200uG * (1.6*BW)^{1/2} \quad (6)$$

Por lo tanto se decidió disminuir este ancho de banda a un valor de 1Hz, este cambio se realizó cambiando los condensadores por unos de 0.47uF. Así se mejoró la resolución y estabilidad de los datos, proporcionando valores más exactos.

Algo importante para resaltar es que se buscó la forma de hacer al inclinómetro autocalibrable, es decir para cuando se desee medir ángulos de inclinación con respecto a una superficie que presente inclinación diferente a donde fue calibrado inicialmente. Pero debido a que en la memoria no se almacenan los datos en posiciones especificas, no es posible identificar cual de los valores se debe recuperar para que sea tomado como un nuevo valor de calibración. Se pensó en tomar los primeros valores leídos por el inclinómetro y sacar un promedio, para tomar este valor para una nueva calibración, no obstante esto fue posible realizarlo únicamente cuando la memoria se encuentra totalmente vacía, lo cual sería una desventaja debido a que los datos deben ser descargados cada vez que se desee utilizar el inclinómetro. En la siguiente tabla se muestran algunas de las especificaciones del inclinómetro.

Tabla 9. Especificaciones

PARÁMETRO	MÍN	MÁX	UNIDADES
Voltaje de Alimentación	7	12	Voltios
Consumo de Corriente	7.4	10.5	mA
Rango de Medición (Región Lineal)	0	±60	Grados(º)
Región no Lineal	±60	±90	Grados(º)
Margen de Error Región Lineal Región no Lineal		 1 5	 Grados (º) Grados (º)

6. OBSERVACIONES Y CONCLUSIONES

- Se diseñó y se implementó un dispositivo que está en capacidad de medir ángulos de inclinación en una superficie, con resolución de un grado en el rango entre -60 y 60 grados. Esta resolución disminuye entre 1 y 5 grados cuando el eje de inclinación se acerca a 90 grados.

- Se experimentó con el microcontrolador Basic Stamp II, el cual fue programado en lenguaje PBASIC. Debido a que PBASIC no maneja interrupciones en su programación (el chip interprete del Basic Stamp 2 es el PIC16C57 el cual no soporta interrupciones), se implementaron subrutinas para realizar las diferentes operaciones.

- Se usó el acelerómetro ADXL202E de tecnología MEMS, aprovechando el hecho que la señal generada a través de sus dos canales de salida es una onda cuadrada con variación de ciclo útil, y de esta forma evitar el uso de etapa de filtrado, amplificación y conversión A/D; lo cual implica una disminución del ruido de cuantización y una notable reducción de tamaño del hardware.

- Se pudo comprobar que una reducción del ancho de banda del acelerómetro por medio de una selección adecuada de los condensadores para los filtros que lo limitan, determinó una reducción de ruido (NOISE FLOOR) y un mejoramiento en la resolución del acelerómetro. Además se obtuvo una mayor estabilidad de los datos.

- Se utilizó un nuevo microcontrolador, el cual presenta una alternativa de programación (lenguaje PBASIC) diferente a los ya conocidos lenguajes

Assembler y C. Otra característica importante es la forma de compilación y descarga del código fuente, la cual es realizada a través de un puerto serial estándar RS232 DB9 sin necesidad de mas circuitos de acople o algún programador adicional.

- Se pudo comprobar que la sensibilidad del acelerómetro ADXL202E es menor a medida que cualquiera de sus ejes efectivos sobrepasa los 80 grados de inclinación, acercándose a los 90 grados. Para aumentar la resolución en el rango de 80º a 90º, la interpolación para determinar las medidas de inclinación se realizó en intervalos más pequeños. Gracias a esto se pudo disminuir considerablemente el margen de error.

- Se pudo comprobar que utilizar un periodo mas corto reduce la resolución del dispositivo, debido a que el número de muestras se disminuyen.

- El microcontrolador híbrido Basic Stamp II no es recomendable para aplicaciones como esta, debido a que se están subutilizando sus características. Por otra parte su costo es elevado con respecto a otros microcontroladores que ofrecen características de funcionamiento suficientes para realizar aplicaciones como la presentada en este documento. Sin embargo se aprovecharon algunas características del Basic Stamp 2 como la reducción en desarrollo de hardware, y con esto no se tuvo la necesidad de implementar etapas de filtrado y amplificación de la señal proveniente del acelerómetro, con esto se reduce la posible presencia de ruido generado en estas etapas

7. RECOMENDACIONES

Si en un futuro se desea continuar con la labor de mejora de las prestaciones de este medidor de inclinación, los autores sugieren tener en cuenta los siguientes aspectos:

- Aumentar el rango de medida a 360 grados y así poder usarlo en medidas de rotación en dos ejes.

- Profundizar más en el lenguaje PBASIC para así poder aprovechar posibles funciones e instrucciones no usadas en el código de programación del medidor de inclinación.

- Diseñar una forma de auto calibración del dispositivo. En este aspecto se trabajo pero sin conseguir buenos resultados, debido a que no se pudo diferenciar y extraer una medida de las demás almacenadas en memoria.

- Realizar otra subrutina en la programación del microcontrolador, como alternativa para que el dispositivo almacene mediciones cada cierto tiempo, y no tener necesidad de estar dependiendo de un operario para el almacenamiento de estas. De esta forma se le da mayor autonomía al dispositivo.

- Diseñar una subrutina que permita la visualización en el LCD del prototipo, de los datos almacenados en la memoria externa, pues con esto no seria necesario un software para la adquisición y visualización de los datos en el PC.

BIBLIOGRAFÍA

[1] ANALOG DEVICES. “ADXL190 Low Cost ± 100 G Single Axis Accelerometer with Analog Output [PDF]” United States Of America 2002

[2] ANALOG DEVICES, “ADXL202E Low-Cost ± 2 G Dual-Axis Accelerometer with Duty Cycle Output [PDF]”, United States Of America 2002

[3] ANALOG DEVICES “ADXL210E Low-Cost ± 10 G Dual-Axis Accelerometer with Duty Cycle Output [PDF]”, United States Of America 2002

[4] ANALOG DEVICES “APPLICATION REFERENCE MANUAL, APPLICATION NOTE AN-214:Ground Rules for High-Speed Circuits, AN-348:Avoiding Passive-Component Pitfalls, AN-280: Mixed Signal Circuit Techniques. Analog Device Inc. 1993.

[5] ANALOG DEVICES. “TECHNICAL NOTE: Mesura d'acceleracions [PDF]”, United States Of America

[6] ANALOG DEVICES, “TECHNICAL NOTE: Using The ADXL202 Duty Cycle Output by Harvey Weinberg [PDF]”, United States Of America 1998

[7] ELECTRONICA PRÁCTICA actual, “Los Basic Stamps: Microcontroladores programables en Basic”, ELECTRONICA PRÁCTICA actual, Año 2-Número 16 Mayo 5 de 2000

[8] MORENO VELASCO Ignacio, Instrumentación electrónica “Apuntes”, Universidad Politécnica de Catalunya, Área de Tecnología Electrónica [PDF].

[9] MIYARA Federico, "El acelerómetro monolítico ADXL05: Un ejemplo de mecatrónica [PDF]"

[10] PARALLAX INC. "BASIC Stamp Programming Manual 2.0c [PDF]" [http:\\] www.parallaxinc.com

[11] PARALLAX INC. "Manual de programación BASIC Stamp 2 Versión 1.1 [PDF]"
[http:\\] www.parallaxinc.com

[12] SCOTT EDWARDS ELECTRONICS INC. "BPI-216 Serial LCD Modules User's Manual Data Sheet" [http:\\] www.seetron.com

[13] SCOTT EDWARDS ELECTRONICS INC. "2X16 Serial LCD Module with Supertwist Display Data Sheet", [http:\\] www.seetron.com

[14] WILLIAMS Jon. "It's All About Angles [PDF]; Column #92 December 2002 "The Nuts and Volts of BASIC Stamps (Volume 3)"

[15] XICOR. "XICOR X25128 (16K x 8Bit) 128K SPI Serial EEPROM with Block Lock™ Protection" [http:\\] www.xicor.com

ANEXOS

ANEXO A. ARQUITECTURA DE LOS BASIC STAMP

Una de las características más importantes de los microcontroladores es su arquitectura, esta muestra como es usada su memoria en cada una de las funciones ejecutadas por el microcontrolador.

El Basic Stamp tiene dos clases de memoria; La RAM (Para las variables usadas por el programa) y la EEPROM (para almacenar el programa, o para almacenar datos de forma similar a la del disco duro en un PC).
Una de las principales diferencias entre la memoria RAM y la EEPROM es que la memoria RAM pierde su contenido cuando el Basic Stamp se desenergiza, por el contrario, la memoria EEPROM no pierde su contenido por mas que se desenergice el Basic Stamp, razón por la cual, en la EEPROM se almacena el código o programa, además de los datos, para el caso en que se necesiten en un futuro.

A.1 ORGANIZACIÓN DE LA RAM EN EL BASIC STAMP 1

El Basic Stamp 1 tiene 16 bytes (8 palabras) de espacio en RAM distribuida como se muestra en la tabla A.1. La primera palabra llamada PORT, es usada para el control de pines de E/S. Esta consiste de 2 bytes, PINS y DIRTS. Cada uno de los bits contenidos en PINS corresponde a cada uno de los ocho pines de E/S en el BS1. Al leer PINS, se leen los pines E/S directamente, retomando en 8 bits puestos a 1's y 0's correspondientes al estado alto y bajo del pin de E/S en ese momento. Escribir a PINS almacenara un valor alto o bajo en los respectivos pines de E/S (únicamente a través de los pines que son puestos como salidas). El segundo byte de PORTS, DIRS, controla la dirección de los pines de E/S.

A.1.1 VARIABLES DE ENTRADA-SALIDA (E/S) (EN EL BASIC STAMP 1)

Las palabras del Basic Stamp 1 son 8 distribuidas como se muestra en la tabla A.1, cada una de las cuales consisten de 2 bytes para un total de 16 bytes. La primera palabra PINS corresponde a cada uno de los ocho pines de E/S en el BS1. La segunda palabra, DIRS contiene en cada uno de sus bits una dirección de un pin de E/S. Un bit alto (1) pone el correspondiente pin de E/S como una dirección de salida y el bit bajo (0) como una dirección de entrada.

Las palabras restantes (W0-W6) están disponibles para uso de propósito general. Cada palabra consiste de bytes direccionables separadamente, y los primeros 2 bytes (B0 y B1) son también bits direccionables.

Se pueden asignar otros nombres (símbolos) a estos registros de la RAM, como se muestra en la sección "definición y uso de variables" que se encuentra mas adelante.

Cuando el Basic Stamp 1 es energizado o reseteado, todas las posiciones de memoria son borradas o puestas a cero, de modo que todos los pines son entradas (DIRS = %00000000). También, si el programa PBASIC pone todos los pines de E/S como salidas (DIRS = %11111111), entonces ellos inicializaran como salidas en bajo, tal que los pines de salida son puestos a cero después de energizar o resetear.

Tabla A.1: Organización de la RAM del Basic Stamp 1

Nombre de palabra	Nombres de Bytes	Nombres de bits	Anotaciones especiales
PORT	PINS DIRS	PIN0 – PIN7 DIR0 – DIR7	Pines I/O; bit direccionable. Direcciones de pines I/O; bit direccionable
W0	B0 B1	BIT0 – BIT7 BIT8 – BIT15	Bit direccionable Bit direccionable
W1	B2 B3		
W2	B4 B5		
W3	B6 B7		
W4	B8 B9		
W5	B10 B11		
W6	B12 B13		Usado por la instrucción GOSUB. Usado por la instrucción GOSUB.

FUENTE: BASIC Stamp Programing Manual 2.0c.

Nota: Hay ocho palabras, consistentes de dos bytes cada una, para un total de 16 palabras. Los bits dentro de las dos palabras más altas son individualmente direccionables.

A.2 ORGANIZACIÓN DE LA RAM EN LOS DEMAS BASIC STAMP (BS2, BS2E, BS2SX, BS2P, BS2PE)

El BS2, BS2E, BS2SX, BS2P y BS2PE tienen 32 bytes de espacio RAM variable, organizado como se muestra en la tabla A.2. Para la cual, los primeros 6 bytes son reservados para entradas, salidas y para el control de dirección de los pines de E/S. Los restantes 26 bytes están reservados para ser usados como de propósito general (como variables).

A.2.1 VARIABLES DE ENTRADA-SALIDA (E/S) (BS2, BS2E, BS2SX, BS2P, BS2PE)

La palabra variable INS es única debido a que es de lectura solamente, los 16 bits de INS reflejan el estado de los pines de E/S, desde P0 hasta P15. Este estado únicamente puede ser leído, no escrito. OUTS contiene los estados de los 16 pines de salida. DIRS controla la dirección (Entrada o salida) de cada uno de los 16 pines de E/S.
Un 0 en un bit particular de DIRS, hace del correspondiente pin una entrada, y un 1 hace del correspondiente pin una salida. De modo que si el bit 5 de DIRS es 0 y el bit 6 de DIRS es 1, entonces el pin 5 (P5) de E/S es una entrada y el pin 6 (P6) de E/S es una salida. Un pin que es una entrada, esta controlado por la circuiteria externa del Basic Stamp; el Basic Stamp no puede cambiar su estado. Un pin que es una salida, es puesto al estado indicado por medio del correspondiente bit del registro OUTS.
Cuando el Basic Stamp es alimentado o reseteado, todas las posiciones de memoria son borradas o colocadas a 0, de modo que todos los pines son entradas (DIRS= %0000000000000000). También, si el programa PBASIC pone todos los pines de E/S como salida (DIRS=

%1111111111111111), entonces se inicializan como una salida baja, debido a que los pines de salida (OUTS) son puestos a cero al energizar o resetear.

Tabla A.2: Organización de la RAM del BS2, BS2e, BS2sx y BS2p.

Nombre de la palabra	Nombres de byte	Nombres de Nibbles	Nombres de bit	Notas especiales
INS	INL INH	INA, INB INC, IND	IN0 – IN7 IN8 – IN15	Pines de entrada
OUTS	OUTL OUTH	OUTA, OUTB OUTC, OUTD	OUT0 – OUT7 OUT8 – OUT15	Pines de salida
DIRS	DIRL DIRH	DIRA, DIRB DIRC, DIRD	DIR0 – DIR7 DIR8 – DIR15	Control de dirección de los pines I/O
W0	B0 B1			
W1	B2 B3			
W2	B4 B5			
W3	B6 B7			
W4	B8 B9			
W5	B10 B11			
W6	B12 B13			
W7	B14 B15			
W8	B16 B17			
W9	B18 B19			
W10	B20 B21			
W11	B22 B23			
W12	B24 B25			

Fuente: BASIC Stamp Programing Manual 2.0c.

Nota: Existen 16 palabras consistentes de dos bytes cada una para un total de 32 bytes.
Todos los bits son individualmente direccionables por medio de un modificador y los bits dentro de las tres palabras mas altas son también individualmente direccionables por medio de nombres pre-definidos.

La variable INS siempre muestra el estado de los pines de E/S, indiferente de la dirección de cada pin de E/S. Si un pin fue puesto en modo de entrada (dentro de DIRS) y un circuito externo conecta al pin de E/S a tierra, el bit correspondiente de INS estaría en bajo. Si un pin fue puesto en modo

salida y el estado del pin fue puesto en nivel alto (dentro de OUTS), el correspondiente bit de INS será alto. Sin embargo, ese mismo pin fue conectado externamente directo a tierra, el correspondiente bit de INS, será bajo. Esto se debe a que se esta leyendo el estado del pin por si mismo y el Basic Stamp no puede pasar por alto un pin que este enviado a tierra o a 5 voltios externamente.

NOTA: El último ejemplo es un error, es un corto directo y puede causar daños al Basic Stamp.

Resumen de la funcion de DIRS, INS y OUTS

DIRS determina si el estado de un pin esta configurado para circuiteria externa (Entrada 0) o por medio del estado del registro OUTS (Salida en 1).
INS siempre indica el estado actual de los pines E/S si son entradas o salidas.
OUTS mantiene los bits que solamente aparecerán en los pines, los cuales son bits en DIRS que están configurados para la salida.

En la programación de la Basic Stamp es recomendable trabajar con bytes, nibbles o bits individuales de INS, OUTS o DIRS en lugar de las palabras completas de 16 bits. PBASIC tiene incluidos nombres para estos elementos, los cuales se muestran en la tabla A.2.
El registro INS es de 16 bits (correspondientes a los pines E/S de 0 hasta 15), este esta compuesto de 2 bytes llamados INL (El byte bajo) y INH (El byte alto). INL corresponde a los pines E/S del 0 al 7 e INH corresponde a los pines E/S del 8 al 15. INS puede estar compuesto por 4 nibbles: INA, INB, INC, e IND. INA corresponde a los pines E/S del 0 al 3, INB corresponde a los pines de E/S del 4 al 7, INC corresponde a los pines E/S del 8 al 11 e IND corresponde a los pines E/S del 12 al 15. Cada uno de los bits de INS puede ser accesado directamente usando los nombres IN1, IN2,…, IN5.
El mismo esquema anterior puede ser usado para las variables OUTS y DIRS.
Como se muestra en la tabla A.2 la memoria del Basic Stamp esta organizada en 16 palabras de 16 bits cada una. Las primeras tres se usan para E/S, las demás 13 se usan como variables de propósito general.
Tal como las variables E/S, las variables de propósito general tienen nombres predefinidos: W0 hasta W12 y B0 hasta B25. B0 es el byte mas bajo de W0, B1 es el mas alto y de la misma forma hasta W12 (B24 es el byte bajo y B25 es el byte alto).
A diferencia de las variables E/S, no hay razón para que las variables de programa tengan que ser restringidas a una posición especifica en la memoria física del Basic Stamp. Un byte, es un byte indiferente de su localización, y si un programa usa una mezcla de variables de diferente tamaño, se hace muy tedioso el almacenamiento y el direccionamiento.
Aun más importante, al mezclar variables con posiciones fijas, con variables localizadas automáticamente, se producen errores de programación (*bugs*). Una variable fija puede traslapar una variable localizada, causando que los datos hechos para una variable se muestren en otras. Los nombres de las variables fijas (De las variables de propósito general), son proveídos únicamente para usuarios avanzados que requieren acceso completo a localizaciones específicas de la RAM.

Se recomienda que evite usar las variables fijas en la mayoría de las situaciones. En lugar de eso, el editor PBASIC localiza las variables.
El software editor organizara los requerimientos de almacenamiento para optimizar el uso de la memoria disponible.

A.3 USO Y DEFINICION DE VARIABLES

Antes de usar una variable en un programa PBASIC, esta se debe declarar.
“Declare”, quiere decir que el BASIC Stamp sepa que se planea usar una variable, que se quiere llamarla, y que tan grande es esta.

Aunque PBASIC tiene variables predefinidas que se pueden usar declarándolas primero, la mejor forma de activar una variable es usar la directiva SYMBOL (para el BS1) o VAR (Para los demás BASIC Stamp).
Esta es la sintaxis para declarar una variable:

En el Basic Stamp 1: SYMBOL Nombre = NombreRegistro

En los demás Basic Stamp: Nombre VAR Tamaño

Donde “Nombre” es el nombre con el cual se refiere a la variable, “NombreRegistro” es el nombre fijo para el registro y “Tamaño” indica el número de bits de almacenamiento para la variable.

A.4 REGLAS PARA LOS NOMBRES DE SYMBOLOS

Hay ciertas reglas con respecto al nombre de los símbolos. Los símbolos deben empezar con una letra, pueden contener una mezcla de letras, números y caracteres subrayados, y deben no ser las mismas palabras claves del PBASIC o etiquetas usadas en el programa. Adicionalmente Símbolo puede ser de una longitud máxima de 32 caracteres. PBASIC no distingue entre mayúsculas y minúsculas.

Para el BS1, el “NombreRegistro” uno de los nombres de variables fijos predefinidos, tales como W0, W1, B0, B1, etc.

Ejemplo de declaración de variable en el Basic Stamp1:

```
SYMBOL   Variable1   = W0        “El valor puede estar entre 0-65535”
SYMBOL   Variable2   = B1        “El valor puede estar entre 0- 255”
SYMBOL   Variable3   = B2        “El valor puede estar entre 0-255”
```

El ejemplo anterior crea una variable llamada “Variable1” cuyo contenido será almacenado en la localización de la RAM llamada W0. De la misma forma, la variable “Variable2” será almacenada en la localización B1 y “Variable3” en B2. “Variable1” es una variable de tamaño palabra, debido a que este es el tamaño de W0. Mientras las otras 2 variables son de tamaño byte. A lo largo del resto del programa, se pueden usar los nombres Variable1, Variable2 y Variable3 en lugar de W0, B1 y B2 respectivamente. Esto hace que el código sea más legible. Esto es más fácil para determinar que “Variable2" es usado de tal forma que este sea la figura que signifique el nombre B1. Ahora “Variable2” reside en la localización B1 y B1 pasa a ser el byte alto de W0. Esto significa que cambiando “Variable2” también cambiara “Variable1”. Una situación similar a esta, usualmente es un error y resulta con resultados desconocidos, sin embargo es una característica poderosa si se usa cuidadosamente.

Para el BS2, BS2e, BS2sx, BS2p y BS2pe, el tamaño discutido tiene 4 secciones:

1) Bit (1 bit)
2) NIB (nibble; 4 bits)
3) Byte (8 bits)
4) Word (16 bits)

Ejemplo de declaración de variable en el BS2, BS2e, BS2sx, BS2p o BS2pe:

```
RATÓN     VAR  BIT     “Valor puede ser 0 o 1”
GATO      VAR  NIB     “El valor puede estar entre 0-15”
```

```
PERRO     VAR  Byte     "El valor puede estar entre 0-255"
CEBRA     VAR  Word     "El valor puede estar entre 0-65535"
```

El ejemplo anterior crea una variable de tamaño bit llamada "RATÓN", una variable de tamaño nibble llamada "GATO", una variable de tamaño byte llamado "PERRO" y una variable de tamaño word llamado "CEBRA". A diferencia del Basic Stamp 1, esta declaración de variables no apunta a una localización específica en la RAM. Por el contrario, solamente se especifica el tamaño deseado para cada variable. El Basic Stamp las ordena en la RAM como van entrando. En el resto del programa, se pueden usar los nombres RATÓN, GATO, PERRO y CEBRA para activar o recuperar el contenido de estas variables.

De una forma general, una variable debería ser dada en el tamaño más pequeño ya que se agarra el valor más grande que pueda ser almacenado en esta. Si se necesita una variable para tomar el estado ON/OFF (1 o 0) de un interruptor, use un bit. Si necesita un contador para un lazo FOR… NEXT que cuente de 1 a 100, use un byte, y así en todo.
Si se asigna un valor para una variable que excede su tamaño, el exceso de bits será perdido. Por ejemplo:
Supóngase que se usa una variable "PERRO" de tamaño nibble, del ejemplo anterior, y escribiendo PERRO = 260 (%100000100 binario).
¿Que podrá contener PERRO?; Este cogerá únicamente los 8 bits menos significativos de 260=%00000100 (4 decimal), el cual es un resultado incorrecto.

A.5 DEFINICION DE *ARRAYS* (ARREGLOS) PARA EL BS2, BS2E, BS2SX, BS2P Y BS2PE

En el BS2, BS2e, BS2sx, BS2p y BS2pe se pueden definir también variables multipartes (o compuestas por varias variables) llamadas *arrays* (arreglos). Un *array* es un conjunto de variables del mismo tamaño, y un solo nombre particionado, pero particionado dentro de celdas numeradas llamadas elementos. La siguiente es la sintaxis para definir un array:

```
Nombre     VAR   Tamaño(n)
```

Donde Nombre y Tamaño tienen las mismas características descritas anteriormente (Ver "USO Y DEFINICIÓN DE VARIABLES"). El nuevo termino (n), le señala al PBASIC cuantos elementos se quiere que tenga el arreglo. Por ejemplo:

```
Libro   VAR  BYTE (10)      'crea un byte con arreglo de 10 elementos'
```

Una vez el *array* esta definido, se puede acceder a estos elementos por medio de un número. La numeración va desde 0 hasta terminar en n-1. Por ejemplo:

```
Libro (3)  =  57
Debug  ?  Libro (3)
```

Este código muestra "Libro (3) = 57" en la pantalla del PC. El fuerte del manejo en *arrays* es que el valor índice puede ser una variable en si. Por ejemplo:

```
Hojas        VAR   BYTE (10)        'crea un byte con arreglo de 10 elementos'
Índice       VAR   NIB              'Define una variable tamaño nibble'

FOR Indice = 0   TO 9               'Repetir con Indice = 0, 1, 2… 9.'
  Hojas (Indice) = Indice * 13      'Escribe Indice * 13 a cada celda del arreglo'
NEXT
```

```
FOR Indice = 0   TO  9              'Repetir con índice = 0, 1, 2... 9.'
  DEBUG   ?   Hojas (Indice)       'Muestra el contenido de cada celda'
NEXT
STOP
```

Si este programa es corrido, el comando DEBUG mostrara cada uno de los 10 valores almacenados en los elementos del *array*: Hojas (0)=0*13=0, Hojas (1)=1*13=13, Hojas (2)=2*13=26,…, Hojas (9)=9*13=117.

Cuidados que se deben tener cuando se trabaja con los *arrays*:

Si se esta familiarizado con otros lenguaje Basic y se tiene que usar uno de estos *arrays*, se tiene que correr dentro del error "subscripción fuera del rango". Suscripción es otro término para el valor del índice (Index). Este es un rango en el que se excede el tamaño de array. En el ejemplo anterior, Hojas es un arreglo de 10 celdas. Los números de índice permisibles son entre 0 y 9, si el programa excede el rango, PBASIC no responde con un mensaje de error. Por el contrario, este accede a la localización de RAM siguiente, antes de finalizar el *array*. Si no se hace esto cuidadosamente, esto puede causar toda clase de errores (*bugs*).
Si acceder a una localización fuera del rango, es malo; ¿Porque PBASIC permite esto?
A diferencia de un computador, el Basic Stamp no tiene siempre una pantalla o un dispositivo conectado para visualizar mensajes de error. De tal manera este es el mejor camino para visualizar esto, esto ayuda a prevenir los *bugs*.
Otra propiedad única de los *arrays* en el lenguaje PBASIC es la siguiente:
Se puede referir a la celda "0" del *array* por medio del uso de solo el nombre del *array*, sin un valor índice.Por ejemplo:

```
Hojas   VAR   BYTE (10)     'Define un byte con arreglo de 10 elementos
Hojas (0) = 17              'Almacena el 17 en la celda cero "0"
DEBUG ? Hojas (0)           'Muestra el contenido de la celda cero "0"'
DEBUG ? Hojas                       'También muestra el contenido de la celda cero "0"'
```

A.6 ALIAS Y VARIABLES MODIFICADORAS

Un alias es un nombre alternativo para una variable existente. Por ejemplo:

En el Basic Samp 1:

```
SIMBOL Gato = B0              'Crea una variable de tamaño byte
SIMBOL Perro = Gato       'Crea otro nombre para la misma variable
```

En los demás Basic Samp:

```
Gato    VAR   BYTE            'Crea una variable de tamaño byte
Perro   VAR   Gatot           'Crea otro nombre para la misma variable
```

En este ejemplo, "Perro" es un alias para la variable "Gato". Algún almacenamiento en "Gato" se mostrará en "Perro" y viceversa. Ambos nombres se refieren a la misma pieza física de RAM. Esta clase de alias pueden ser usados cuando se necesite volver a usar una variable temporalmente en una diferente posición en un programa, sin embargo también se requiere el nombre de la variable para repetir esta función en cada sitio. Se debe tener cuidado al trabajar con alias por que esto es fácil de olvidar; durante la ejecución del programa se preguntara ¿Cómo este valor se encuentra aquí? la respuesta es que esta fue almacenada en el alias de la variable.

En el BS2, BS2e, BS2sx, BS2p, y BS2pe un alias también puede servir como una ventana dentro de una porción de otra variable. Esto es posible usando "modifiers" (modificadores).

Aquí el alias es asignado con un modificador que especifica que parte:

```
Cebra   VAR     WORD                    'Una variable de 16 bits
Cabeza  VAR     Cebra.HIGHBYTE          'Los 8 bits más altos de Cebra
Cuerpo  VAR     Cebra.LOWBYTE           'Los 8 bits más bajos de Cebra
```

Por ejemplo, si se escribe el valor %1011000011111101 para "Cebra", Entonces "Cabeza" contendrá %10110000 y "Cuerpo" contendría %11111101.
La tabla A.3 lista todas las variables modificadoras. PBASIC deja que se le apliquen estos modificadores a cualquier nombre variable y combinarlos en cualquier forma que tenga sentido. Por ejemplo:

```
Cebra   VAR     WORD                            'Una variable de 16 bits
Boca    VAR     Cebra.HIGHBYTE.LOWNIB.BIT1      'Un bit
```

Tabla A.3: Variables modificadoras del BS2, BS2e, BS2sx, BS2p y BS2pe.

Símbolo	Definición
LOWBYTE	Bit bajo de una palabra
HIGHBYTE	Bit alto de una palabra
BYTE0	byte 0 (byte bajo) de una palabra
BYTE1	byte 1 (byte alto) de una palabra
LOWNIB	Nibble bajo de una palabra o byte
HIGHNIB	Nibble alto de una palabra o byte
NIB0	nib 0 de una palabra o byte
NIB1	nib 1 de una palabra o byte
NIB2	nib 2 de una palabra
NIB3	nib 3 de una palabra
LOWBIT	Bit bajo de una palabra ,byte o nibble
HIGHBIT	Bit alto de una palabra, byte o nibble
BIT0	bit 0 de una palabra, byte o nibble
BIT1	bit 1 de una palabra, byte o nibble
BIT2	bit 2 de una palabra, byte o nibble
BIT3	bit 3 de una palabra, byte o nibble
BIT4 ... BIT7	bits 4 al 7 de una palabra o byte
BIT8 ... Bit15	bits 8 al 15 de una palabra

Fuente: **BASIC Stamp Programing Manual 2.0c.**

Una regla de sentido común para combinar modificadores es que se deben usar progresivamente desde el más pequeño de derecha a izquierda. También se puede usar para definir o indicar la posición de una parte de la variable. Ejemplo:

```
Cebra     VAR     WORD          'Una variable de 16 bits
Boca      VAR     Rhino.BIT9    'Un bit
```

Usando en instrucciones de programas:

```
Cebra     VAR     WORD                  'Una variable de 16 bits
Cabeza    VAR     Cebra.HIGHBYTE    '8 bits más significativos de Cebra

Cebra = 13567
DEBUG  ? Cabeza                   'Muestra el valor del alias de la variable Head'
DEBUG? Cebra.HIGHBYTE       'Trabaja también Cebra.HIGHBYTE'
STOP
```

Los modificadores también trabajan con *arrays.* Por ejemplo:

```
Hojas     VAR   BYTE (10)              'Define un array de 10 bytes
Hojas(0) = $AB                      'Hex$AB dentro del Byte 0
DEBUG HEX ? Hojas.LOWNIB (0)      'Muestra el nib mas bajo ($B)
DEBUG HEX ¿ Hojas.LOWNIB(1)       'Muestra el nib mas alto ($A)
```

Esta propiedad de modificar *arrays* hace los nombres un poco confusos. Si se prefiere, se puede usar las versiones sin descripción de los nombres modificadores; BIT0 en lugar de LOWBIT, NIB0 en lugar de LOWNIB, y BYTE0 en lugar de LOWBYTE. Estos tienen exactamente el mismo efecto, pero quizá es menos probable que sea mal interpretado.

Se pueden usar modificadores con la celda 0 de un *array* referenciando solamente el nombre del *array* sin el valor del índice en paréntesis. Esto es presa fácil para los alias y modificadores, ambos en directivos VAR y en instrucciones.

A.7 EXPRESIONES CONSTANTES Y TIEMPO-COMPILAR

Se llaman constantes a cualquier valor que mientras el programa esta corriendo, nada puede hacerse para cambiar su valor. Esto distingue las constantes de las variables, las cuales pueden cambiar mientras el programa esta corriendo.
PBASIC permite el usar varios sistemas de numeración. Por defecto este asume que los números están en decimal (base 10), nuestro sistema numérico. Pero también se pueden usar números binarios y hexadecimales, estos se identifican con prefijos (Ver "REPRESENTACION NUMÉRICA").
Por ejemplo:

```
99          'Decimal'
%1010       'Binario'
%FE         'Hexadecimal'
"A"         'ASCII  para A (65)'
```

Se puede asignar nombres a las constantes de forma similar a como las variables están declaradas. En un Basic Stamp 1, se hace de forma idéntica a la declaración de variables y en los demás Basic Stamp se usa el directivo CON. Esta es la sintaxis.

En el Basic Stamp 1:

```
SIMBOL       Nombre =  Valor de la constante
```
En los demás Basic Stamp:
```
Nombre       CON       Valor de la constante
```

Una vez declaradas, pueden ser usadas en lugar de los números que ellos representen:

En el Basic Stamp 1:

```
SIMBOL       Casa  = 3              'Valor numérico de Casa

FOR     Contador = 1 TO Casa
  GOSUB  Lista
NEXT
…
```

En los demas Basic Stamp:

```
Casa      CON    3                      'Valor numérico de Casa

FOR   Contador  = 1 TO Casa
   GOSUB  Lista
NEXT
...
```

Este código puede trabajar exactamente igual a los anteriores FOR…NEXT (Lazos). El software editor sustituirá el número 3 por la constante llamada Casa para todo el programa. Al igual que los nombres de variables, etiquetas e instrucciones, el Basic Stamp no diferencia mayúsculas de minúsculas en los nombres de constantes, es decir:
CASA y CaSa serán procesadas por el Basic Stamp como Casa.

Usar constantes nombradas no incrementa la cantidad de código descargado al Basic Stamp, y este a menudo mejora la claridad del programa.
La declaración de constantes también tiene otros beneficios. Por ejemplo:
Si se necesita que para el ejemplo anterior el valor constante no sea 3 sino 7, entonces en el ejemplo se deben cambiar todos los 3's por 7's. Buscar y remplazar ayudara, pero puede accidentalmente cambiar algunos 3's que no fueron correspondan a el valor de Casa, lo cual llevara a alterar la eficiencia del código. Pero si la constante fue declarada con anterioridad se puede cambiar en un solo paso todos los 3's por 7's. Así:

En el Basic Stamp 1:

```
SYMBOL          Casa = 7                    'Nuevo valor numérico de Casa
```

En los demas Basic Stamp:

```
Casa       CON    7                        'Nuevo valor numérico de Casa
```

Otra característica importante de la en la declaración de constantes, es que una constante se puede definir en términos de otra. Por ejemplo:

```
Casa           CON       7
Casona         CON       Casa+2
```

Como se puede ver, una constate puede ser definida en términos de otra, es decir que la constante Casona depende del valor de la constante Casa.

Se puede usar el ejemplo anterior para definir expresiones con constantes; los grupos de operaciones matemáticas y/o lógicas que el software editor resuelve (evalúa) a tiempo de copilado (Tiempo justo después de empezar la descarga y antes que el Basic Stamp comienza a correr el programa).
Las expresiones usadas para definir constantes se deben mantener simples. El software editor resuelve de izquierda a derecha y no permite usar paréntesis para cambiar el orden de la evaluación. Los operadores permitidos en expresiones constates son mostrados en la tabla A.4.

Tabla A.4: Operadores permitidos en expresiones constantes para el BS2, BS2e, BS2sx, BS2p y BS2pe.

Operador Símbolo	Descripción
+	Suma
-	Resta
*	Multiplicación
/	División
<<	Moverse a la izquierda (Shift Left)
>>	Moverse a la derecha (Shift Right)
&	AND Lógico
|	OR Lógico
^	XOR Lógico

Fuente: **BASIC Stamp Programing Manual 2.0c.**

El Basic Stamp, como cualquier computador, tiene buen desempeño en matemática y lógica. Sin embargo, siendo designado para aplicaciones de control, el Basic Stamp matemáticamente tiene una pequeña diferencia en desempeño a una calculadora.

A.8 REPRESENTACIÓN NUMÉRICA

En un programa se puede expresar un número de varias formas, dependiendo de cómo será usado el número y que significado tiene. Por defecto, el Basic Stamp reconoce los números 10, 99 o 62145 como números decimales (En base 10). Sin embargo también pueden ser usados números hexadecimales (En base 16), o binarios (En base 2). Para evitar que números comunes (Como 0 y 1) entre los diferentes sistemas numéricos no sean interpretados de forma correcta por el Basic Stamp, el software editor necesita prefijos para nombrar de forma diferente los sistemas numéricos. Para representar de forma correcta los diferentes sistemas numéricos, se tienen los siguientes prefijos:

1. Los números decimales no necesitan de ningún prefijo, el Basic Stamp cualquier número que no tenga prefijo asume que es decimal.
2. A los números hexadecimales se les antepone el prefijo $.
3. A los números binarios se les antepone el prefijo %.

El Basic Stamp también convierte automáticamente el texto citado en código ASCII, y permite aplicar nombres (símbolos) a constantes de cualquiera de los sistemas de numeración, por ejemplo:

En el Basic Stamp 1:

```
SYMBOL      Letra A   =   "A"          'Codigo ASCII para A (65)
SYMBOL      Casa      =    3
SYMBOL      Hex128    =    $80
SYMBOL      FewBits   =   %1101
```

En los demás Basic Stamp:

```
Letra A CON    "A"              'Codigo ASCII para A (65)
Casa    CON    3
Hex128  CON    $80
FewBits        CON    %1101
```

A.9 RUNTIME Y TIEMPO DE COMPILACION

En los Basic Stamp, no todas las operaciones lógicas y matemáticas en un programa son resueltas por este. El software editor resuelve operaciones que definen constantes antes que el programa es descargado en el Basic Stamp, y a este preprocesamiento el cual tiene lugar antes que el programa sea descargado es llamado "tiempo de compilación".

Después la descarga es completada, el Basic Stamp empieza ejecutando el programa "runtime". En runtime el Basic Stamp procesa operaciones matemáticas y lógicas involucrando variables, o cualquier combinación de variables y constantes.
Sin embargo el tiempo de compilación y la expresión runtime parecen similares. Un pequeño ejemplo puede ayudar:

```
Resultado       VAR      BYTE     'Tiempo de compilación asignado
Casa            CON      3                'Tiempo de compilación
Puerta  CON      Casa* 2-1          'Tiempo de compilación
Ventana         CON      100 +90            'Tiempo de compilación'
Piso            CON      3*Resultado      'ERROR: variables no permitidas'
Resultado = Puerta          'Runtime'
Resultado = 99+Puerta       'Runtime'
Resultado = Ventana + 1          '"100+90" resuelto en el tiempo de compilación'
                                 'Ventana + 1  resuelto en el runtime'
Resultado = 100+90          '100+90  resuelto en el runtime'
```

Note que el último ejemplo es resuelto en el runtime, aun cuando la matemática ejecutada tenga que ser resuelta en el tiempo de compilación, debido a que este incluye dos constantes. Si se encuentra algo como esto en el programa, se puede guardar algún espacio EEPROM para convertir la expresión runtime 100+90 dentro de la expresión de tiempo de compilación Puerta con 100+90.

Las expresiones de tiempo de compilación son aquellas que involucran únicamente constantes; una vez la variable es involucrada, la expresión debe ser resuelta al ejecutar el programa (runtime)

esto es porque la línea “Piso CON 3*Resultado” generaría un mensaje de error. La directiva CON trabaja únicamente en el tiempo de compilación, debido a que las variables no son permitidas.

A.10 OPERACIONES ARITMETICAS

En las operaciones aritméticas como la suma, resta, multiplicación y división, se debe tener en cuenta que el orden en el cual se desarrolla la operación no afecte el resultado. Cuando se suma o se resta el orden de los factores no alteran el resultado, sin embargo en la multiplicación y en la división se deben usar paréntesis alrededor de las porciones involucradas en estas operaciones.

El Basic Stamp resuelve problemas matemáticos en el orden en que son escritos; de izquierda a derecha. El resultado de cada operación es alimentado dentro de la siguiente operación. Así al computar 10+3*2/4, el Basic Stamp la desarrolla en el siguiente orden:

10 + 3 = 13
13 * 2 = 26
26 / 4 = 6

El Basic Stamp solo desarrolla matemática de números enteros. Por esto 26/4 da como resultado 4 y no 6.5.

El Basic Stamp 1 no permite paréntesis en las expresiones. Desafortunadamente todas las expresiones deben ser escritas en una forma estricta, ya que estas se avaluaran de izquierda a derecha.
Los demás Basic Stamp, permiten el uso de paréntesis para cambiar el orden de evaluación. Al encerrar una operación matemática en paréntesis se le da prioridad sobre otras operaciones. El Basic Stamp 2 trabaja de forma tradicional es decir: para realizar la operación 12+ (3*2/4), primero realiza la operación dentro del paréntesis de izquierda a derecha y luego suma el resultado con el 12.
En caso de tener varios paréntesis dentro de otro paréntesis (paréntesis anidado), el Basic Stamp resuelve desde el paréntesis más interno hacia fuera.

A.11 MATEMATICA ENTERA

El Basic Stamp resuelve las operaciones matemáticas por la regla de matemática entera positiva, esto es; que solamente manipula números enteros y elimina cualquier porción fraccional de los resultados. El Basic Stamp manipula números negativos usando reglas de complementa a 2.
Los Basic Stamp (Menos BS1) pueden interpretarlos números negativos en complemento a 2 correctamente en las instrucciones DEBUG Y SEROUT usando modificadores como SDEC (para decimales con signo). En cálculos, sin embargo, asume que todos los valores son positivos. Esto da resultados correctos para adiciones de números negativos en complemento a 2, también para la substracción y la multiplicación, pero no para la división.

A.12 OPERADORES UNITARIOS Y BINARIOS

Las operaciones estándar (Suma, resta, multiplicación y división) todas trabajan sobre dos valores; como en 2+3 o 5*14. Los valores que lo operadores procesan son tomados como argumentos de tal manera que los operadores estándar procesan dos argumentos.
Los operadores que toman un argumento son llamados unitarios, y los que toman dos binarios (binario no tiene nada que ver con los números binarios).

El signo menos (-) es un bit de un híbrido, puede ser usado como un operador unitario como el caso de -8.
Los operadores lógicos y matemáticos de dividen en dos tipos unitarios y binario, los unitarios tienen prioridad sobre los binarios, siempre las operaciones unitarias son resueltas primero. Por ejemplo, SQR es el operador unitario para la raíz cuadrada, en la expresión 10-SQR 16, el Basic Stamp toma primero la raíz de 16 y luego la substrae del 10.

Notas acerca del area de trabajo de 16 bits

Muchas de las descripciones que se emplean son algo como 'compute (*some function*) de un valor de 16 bits'. Esto significa que el computo es realizado en un *workspace* de 16 bit. Si el valor es mas pequeño de 16 bits, el Basic Stamp coloca ceros para hacer este valor de 16bits. Si los 16 bits resultan de un cálculo, y son para ser empaquetados en una variable más pequeña, los bits mas significativos son truncados.

Por ejemplo, como hacer para que el número -99 llegue a ser 157. Observemos que 99 es %01100011 en binario. Para que el Basic Stamp niegue el 99 este primero se convierte en un número de 16 bits (%0000000001100011) y después se le hace el complemento a dos (%1111111110011101). Entonces para poder colocar el resultado en una variable de 8bits, los ocho bits más significativos son truncados y los 8 menos significativos son almacenados en el byte: % 10011101.
Ahora para la segunda parte, el modificador SDEC del DEBUG espera un valor de 16 bits con complemento a dos, pero nosotros solo le damos un byte con el cual trabajar. Como es usual, se crea un valor de 16 bits agregando ceros en los bits más significativos: %0000000010011101, cuyo correspondiente número en decimal es 157.

TABLA A.5: OPERADORES UNITARIOS

Operador	Descripción	Realizado por:
ABS	Valor absoluto	Todos excepto BS1
COS	Coseno en radianes binarios de complemento a dos	Todos excepto BS1
DCD	Decodificador de potencias 2^{x}	Todos excepto BS1
~	Inverso	Todos excepto BS1
-	Negativo	Todos
NCD	Prioriza la codificación de un valor de 16 bits	Todos excepto BS1
SIN	Seno en radianes binarios de complemento a dos	Todos excepto BS1
SQR	Ruiz cuadrada	Todos excepto BS1

Fuente: BASIC Stamp Programing Manual 2.0c.

VALOR ABSOLUTO (ABS)

El operador de valor absoluto (ABS) convierte un número de 16 bit con signo (con complemennto a dos) en su valor absoluto. El valor absoluto de un número, es un número positivo que representa la diferencia entre este número y 0.
Por ejemplo: el valor absoluto de -99 es 99 y el valor absoluto de 99 también es 99. ABS trabaja sobre números negativos de complemento a dos.
Ejemplo:

```
Resultado       VAR   WORD
Resultado = -99
DEBUG SDEC ? Resultado.
DEBUG SDEC ?  ABS  resultado
```

COSENO (COS)

El operador coseno halla el coseno de un número de 16 bits en complemento a dos, como un ángulo de 8 bits (0 a 255). El SIN (seno) y coseno son lo mismo en todos los aspectos, excepto que la función coseno usa la distancia X en lugar de la distancia Y.

DECODIFICADOR (DCD)

El operador decodificador es un decodificador de potencias 2^n de valores de 4 bits; DCD acepta un valor de 0 a 15, y lo convierte en un número de 16 bits, con el bit que describe el valor (bit n = 12) en 1.

```
Resultado    VAR  WORD
Resultado =  DCD 12              'Pone el bit n = 12 en 1
DEBUG  BIN?  Resultado           'muestra el resultado (%0001000000000000)
```

INVERSO (~)

El operador inversor complementa (invierte) los bits de un número. Cada bit que contienen un uno (1) es convertido a 0 y cada bit que contiene un cero es convertido a uno (1), este proceso es también conocido como un " *bitwise NOT*" y complemento a uno. Por ejemplo:

```
Resultado       VAR  BYTE
Resultado =  % 11110001          'almacene en el byte resultado
DEBUG BIN ? Resultado            'muestra en binario (%11110001)
Resultado = ~ Resultado                  'hace el complemento
DEBUG BIN? Resultado                     'muestra en binario (%00001110)
```

NEGATIVO (-)

El operador negativo (-) niega un número de 16 bits (convierte a complemento a dos). Ejemplo:

```
EN EL BASIC STAMP 1
SYMBOL   Resultado = W1
Resultado  =  -99        'pone -99 (en complemento a dos)  dentro de Resultado
Resultado =  Resultado+100          'Suma 100'
DEBUG Resultado                             'muestra Resultado (1)
```

En los demas Basic Stamp:

```
Resultado       VAR    WORD
Resultado= -99      'pone -99 (en complemento a dos)  dentro de Resultado
DEBUG SDEC?  Resultado   'Muestrelo en pantalla como un número con signo
Resultado = -Resultado          'Niegue el valor'
DEBUG SDEC?   Resultado  'muestrelo en pantalla como un número con signo
```

CODIFICADOR (NCD)

El operador codificador (NCD) es un codificador "prioritario" de valores de 16 bits. NCD toma valores de 16 bits, encuentra el bit mas significativo que contenga un 1 y le suma uno (1 a 16), si ningún bit esta en uno (el valor de la entrada es cero) NCD lo deja como cero. NCD es una rápida forma de obtener la respuesta a: ¿Cuál es la potencia más grande en base 2 en este valor? La respuesta a esta pregunta será la potencia mas uno.
Ejemplo:

```
Resultado    VAR     WORD
Resultado = %1101                'El bit mas significativo es el tres'
DEBUG  ?   NCD   Resultado     'Muestre el resultado del NCD (4)'
```

SENO (SIN)

El operador seno (SIN) se halla con el complemento a dos, como una seno de ángulo de 16 bits a partir de un ángulo especificado de 8 bits (0 a 255).
Para entender el operador seno observaremos una típica función seno. Por definición: dado un circulo con radio de 1 unidad (conocido como un circulo unitario), el seno es la distancia proyectada sobre el eje coordenado "Y" desde un punto de la circunferencia ,el cual se obtiene por un ángulo dado medido desde el centro de la circunferencia. Los ángulos son medidos con relación a las 3 en punto, incrementando en contra de las manecillas del reloj.
En el punto de origen (0 grados) el seno es 0, porque ese punto tiene la misma vertical coordenada Y que el centro del circulo. A 45 grados el seno es 0.707. A 90 grados, el seno es 1. A 180 grados el seno es cero de nuevo. A 270 grados el seno es -1.

El operador seno del Basic Stamp divide el círculo en unidades de 0 a 255 en lugar de 0 a 359 grados.
Algunos textos llaman esta unidad como "Radian Binario" o *brads*.
Cada radian binario equivale a 1.406 grados. Y en lugar de un círculo unitario del cual resulta valores seno entre cero y uno, el seno del Basic Stamp esta basado en un circulo de 127 unidades.
Los resultados son dados en complemento a dos en orden para acomodar valores negativos.
En el origen SIN es 0; a 45º (32 *brads*) el seno es 90; a 90º (64 *brads*), el seno es 127, a 180 (128 *brads*), el seno es 0, y a 270 (192 *brads*) el seno es -127.
Para convertir *brads* a grados, se multiplica por 180 y luego se divide por 128. Para convertir grados a *brads*, se multiplica por 128 y luego se divide por 180.
Ejemplo:

```
Casa   VAR   WORD                        'Define variables
Seno   VAR    WORD
FOR   Casa = 0 TO 359 STEP 45            'Usa grados
Seno = SIN (Casa*128/180)                'Convieerte a brads y hace el seno
Debug "Angulo:", DEC Casa, TAB, "Seno", SDEC Seno, CR      'Mostrar
NEXT
```

RAIZ CUADRADA (SQR)

El operador raíz cuadrada (SQR) computa la raíz cuadrada entera de un número de 16 bits (el número debe ser positivo, debido a que la raíz de un número negativo es un número imaginario). Recuerde que muchas de las raíces cuadradas tienen una parte fraccional que el Basic Stamp descarta cuando hace su matemática de solo enteros. Por ejemplo se computa la raíz cuadrada de 100 como 10, pero la raíz cuadrada de 99 se computa como 9 (siendo 9.95 el valor aproximado).Ejemplo:

```
DEBUG SQR 100      'Muestre la raíz cuadrada de 100
DEBUG SQR 99       'Muestre la raíz cuadrada de 99
```

Tabla 4.6: Operadores binarios

Operador	Descripción	Realizado por:
+	Adición	Todos
-	Substracción	Todos
*	Multiplicación	Todos
**	Multiplicación (toma los 16 bits mas significativos)	Todos
*/	Multiplicación por un numero entero de 8-bits y una fracción de 8-bits.	Todos excepto BS1
/	División	Todos
//	Modulo (Residuo de división)	Todos
MIN	Limita un valor a un bajo especificado	Todos
MAX	Limita un valor a un alto especificado	Todos
DIG	Halla el digito especificado de un numero.	Todos excepto BS1
<<	Desplaza los bits hacia la izquierda una cantidad de posiciones especificadas.	Todos excepto BS1
>>	Desplaza los bits hacia la derecha una cantidad de posiciones especificadas.	Todos excepto BS1
REV	Espejo de un numero de bits especificados.	Todos excepto BS1
&	AND	Todos
|	OR	Todos
^	XOR	Todos
&/	AND NOT Lógico	Solo BS1
|/	OR NOT Lógico	Solo BS1
^/	XOR NOT Lógico	Solo BS1

Fuente: BASIC Stamp Programing Manual 2.0c.

Nota: No todos los operadores binarios son llevados a cabo por todos los BASIC Stamp

SUMA (+)

El operador adición, suma variables y/o constantes, convirtiendo el resultado en un número de 16 bits. Trabaja como era de esperarse, con enteros sin signo desde 0 a 65535. Si el resultado de la suma es mas grande que 65535, el bit de acarreo será perdido. Si los valores sumados son números de 16 bits, la destinación es una variable de 16 bits.
Ejemplo:

En el Basic Stamp 1:

```
SYMBOL  Valor1 = W0
SYMBOL  Valor2 = W1
Valor1 = -99
Valor2 = 100
Valor1 = Valor1 + Valor2          'Suma los números
DEBUG  Valor1                     'Mostrar el resultado (1)
```

En los demás Basic Stamp:

```
Valor1 VAR WORD
Valor2 VAR WORD
Valor1 = -1575
Valor2 = 976
Valor1 = Valor1 + Valor2        'Suma los números
DEBUG  SDEC  ?  Valor1          'Mostrar el resultado (-599)
```

RESTA (-)

El operador resta (-) substrae variables y/o constantes, convirtiéndolo en un resultado de 16 bits. Trabaja exactamente como se esperaría, con entero no definidos desde 0 a 65535. Si el resultado es negativo, será correctamente expresado como un número de 16 bits. Por ejemplo
En el Basic Stamp 1:

```
SYMBOL  Valor1 = W0
SYMBOL  Valor 2= W1
Valor1 = 199
Valor2 = 100
Valor1= Valor1 - Valor2       'reste los números
DEBUG    Valor1                     'mostrar el resultado (99)
```

En los demás Basic Stamp:

```
Valor1  VAR    WORD
Valor2          VAR    WORD
Valor1 = 1000
Valor2 = 1999
Valor1 = Valor1 - Valor2                'reste los números
DEBUG SDEC ?  Valor1          'muestre el resultado (-999)
```

MULTIPLICADOR (*)

El operador multiplicador, multiplica variables y/o constantes, volviendo el resultado los 16 bits más bajos. Trabaja con enteros sin definir (signo) desde 0 a 65535. Si el resultado de la multiplicación es más grande de 65535, el exceso de bits se perderá. La multiplicación de variables signadas será correcta (en número y signo), previendo que este resultado esta dentro del rango de -32767 a 32767.
Ejemplo:

En el Basic Stamp 1:

```
SYMBOL  Valor1= W0
SYMBOL  Valor2 = W1
Valor1 = 1000
Valor2 = 19
Valor1= Valor1 * Valor2          'multiplica el valor uno por el valor dos
DEBUG Valor1                     'muestra el resultado (19000)
```

En los demás Basic Stamp:

```
Valor1  VAR     WORD
Valor2          VAR     WORD
Valor1 = 1000
Valor2 = -19
Valor1 = Valor1 * Valor2                 'multiplica los números
DEBUG SDEC ?  Valor1                     'muestra el resultado (-19000)
```

MULTIPLICADOR ALTO (**)

El operador de multiplicador alto, multiplica variables y/o constantes, vuelve el resultado los 16 bits más altos.
Cuando se multiplican 2 números de 16 bits el resultado puede ser tan largo como 32 bits. Ya que la variable más grande que es soportada por el Basic Stamp es de 16 bits, los otros 16 bits son perdidos. La instrucción de doble estrella (**) da los 16 bits mas significativos. La instrucción estrella (*) retornaría a los 16 bits mas bajos.
Por ejemplo, Si se multiplica 65000 ($FDE8) por si mismo. El resultado sera 4.225.000.000 o $FBD46240. El operador multiplicador (*) Tomará los 16 bits mas bajos ($6240), mientras que el operador multiplicador alto (**), tomará los 16 bits mas altos ($FBD4). Ejemplo:

En el Basic Stamp 1:

```
SYMBOL  Valor1= W0
SYMBOL Valor2 = W1
Value1 = $FDE8
Value1= Valor1 ** Valor2         'multiplica el valor uno por si mismo
DEBUG $Valor1                    'muestra el resultado (16 bits mas altos)
```

En los demás Basic Stamp:

```
Valor1  VAR     WORD
Valor2          VAR     WORD
Valor1 = $FDE8
Valor1 = Valor1 ** Valor2                'multiplica los números
DEBUG  HEX ?  Valor1             'muestra el resultado (16 bits mas altos)
```

MULTIPLICADOR MEDIO (*/)

El operador multiplicador medio (*/) multiplica variables y/o constantes, vuelve el resultado los 16 bits intermedios de un resultado de 32 bits. Esto se usa para multiplicar un valor por un número entero y una fracción. El número entero es el byte mas alto de el multiplicador (0 a 255 unidades) y la fracción es el byte mas bajo de el multiplicador (0 a 255 unidades de cada 1/126).
La instrucción (*/, "*star-slash*") proporciona un excelente manejo de la matemática de solo enteros del Basic Stamp. Si se multiplica un valor por 1,5, el byte mas alto del multiplicador será 1, y el byte mas bajo será 128 (parte fraccional), debido a que 128/256=0,5; de tal manera, si se expresa el multiplicador 1,5 en hexadecimal, quedaría como $0180 donde 01 es la parte entera y 80 la parte fraccional, debido a que hex mantiene el contenido de el byte mas alto y bajo por separado. Ejemplo:

En los demás Basic Stamp:

```
Valor1   VAR   WORD
Valor1= 100
Valor1= Valor1*/ $0180          ' Multiplica por 1.5 [1 + (128/256)]
DEBUG ? Valor1                  ' muestra el resultado (150)
```

DIVISION (/)

El operador división divide variables y/o constantes convirtiendo el resultado en un número de 16 bits. Trabaja con números enteros sin signo de 0 a 65535. Se debe usar la división solo con números positivos; los valores con signo no proveen resultados correctos. Ejemplo:

En el Basic Stamp 1:

```
SYMBOL Valor1 = W0
SYMBOL Valor2 = W1
Valor1= 1000
Valor2= 5
Valor1= Valor1 / Valor2          ' Divide los números
DEBUG Valor1                     ' muestra el resultado (200)
```

En los demás Basic Stamp:

```
Valor1   VAR   WORD
Valor2   VAR   WORD
Valor1= 1000
Valor2= 5
Valor1= Valor1 / Valor2          ' Divide los números
DEBUG DEC ? Valor1               ' muestra el resultado (200)
```

Una forma para habilitar la división con números con signo es tener un programa para dividir valores absolutos, después negar el resultado si uno de los operandos es de signo negativo.

MODULO (//)

Recupera el residuo después de la división de un valor en otro. Algunos problemas de división no tienen un número entero como resultado, sino un número y una fracción. Por ejemplo 1000/6=

166,667. La matemática entera no permite la porción fraccional en el resultado, entonces 1000/6 = 166, la cual es la respuesta aproximada, porque 166*6 =996. La operación división deja un residuo de 4.
El (//) "doble *slash*" recupera el residuo de una operación división dada.

En el Basic Stamp 1:

```
SYMBOL Valor1 = W0
SYMBOL Valor2 = W1
Valor1= 1000
Valor2= 6
Valor1= Valor1 // Valor2          ' Obtiene el residuo de Valor1 / Valor2
DEBUG Valor1                      ' Muestra el resultado  (4)
```

En los demás Basic Stamp:

```
Valor1   VAR    WORD
Valor2   VAR    WORD
Valor1= 1000
Valor2= 6
Valor1= Valor1 // Valor2           ' Obtiene el residuo de Valor1 / Valor2
DEBUG DEC ? Valor1                 ' Muestra el resultado  (4)
```

MINIMO (MIN)

El operador mínimo (MIN), limita un valor para un número positivo especificado mínimo de 16 bits. La sintaxis del mínimo es: "Valor MIN limite", donde "valor" es un valor constante o variable sobre la cual se desempeña la función mínimo y "limite" el mínimo valor que puede tomar "valor".
Su lógica es: Si "valor" es menor que "limite", luego haga "resultado = limite"; si "valor" es mas grande que, o, igual a "limite", haga "resultado = valor".
MIN trabaja en matemática positiva solamente, sus comparaciones no son validas cuando son usadas sobre números negativos complemento a dos, debido a que la representación entera positiva de un número como -1 ($FFF o 65535 en decimal sin signo) es mas grande que la de un número como 10 ($00A o 10 decimal). MIN debe ser usado únicamente con enteros sin signo. Ejemplo:

En el Basic Stamp 1:

```
SYMBOL Valor1 = W0
SYMBOL Valor2 = W1
FOR Valor1= 100 TO 0 STEP -10   ' pasa el valor de Valor1 desde 100 hasta 0
Valor2 = Valor1 MIN 50          ' Usa MIN para fijarlo a 50
DEBUG Valor2                    ' muestra el valor "fijado"
NEXT
```

En los demás Basic Stamp:

```
Valor1   VAR    WORD
FOR Valor1= 100 TO 0 STEP 10    ' pasa el valor de Valor1 desde 100 hasta 0
DEBUG ? Valor1 MIN 50           ' muestra Valor1, pero usa MIN para fijarlo a 50
```

```
NEXT
```

MAXIMO (MAXIMO)

El operador máximo (MAX) limita un valor a un número positivo máximo de 16 bits. La sintaxis de MAX es igual a la del operador mínimo; únicamente cambia la palabra MIN por MAX, además conserva sus características.

```
Valor   MAX  limite
```

Su lógica es: "si "Valor" es mas grande que "limite" entonces haga "resultado=limite"; si "Valor" es menor que o igual a "limite", haga: resultado= Valor".
MAX trabaja solamente en matemática positiva. Sus comparaciones no son validas cuando se usan números negativo con complemento a dos, ya que la representación entera positiva de un número como -1 es más grande que la de un número como 10. Se usa MAX solo con enteros positivos. Ejemplo:

En el Basic Stamp 1:

```
SYMBOL Valor1 = W0
SYMBOL Valor2 = W1
FOR Valor1= 0 TO 100 STEP 10   ' pasa el valor de Valor1 desde 100 hasta 0
Valor2 = Valor1   MAX   50          ' Usa MIN para fijarlo a 50
DEBUG Valor2                        ' muestra el valor "fijado"
NEXT
```

En los demás Basic Stamp:

```
Valor1   VAR   WORD
FOR Valor1= 0 TO 100 STEP 10   ' pasa el valor de Valor1 desde 100 hasta 0
DEBUG ? Valor1  MAX  50   'muestra Valor1, pero usando MIN para fijarlo a 50
NEXT
```

DIGITO (DIG)

El operador digito (DIG) recupera un digito decimal especificado de un valor positivo de 16 bits. Los dígitos son numerados desde 0 (digito mas a la derecha) hasta 4 (el digito mas a la izquierda de un número de 16 bits; 0 a 65535).Ejemplo:

En los demás Basic Stamp:

```
Valor       VAR   WORD
Todos       VAR   BYTE
Valor = 9742
DEBUG ? Valor DIG 2          ' Muestra el digito 2 (7)
FOR Todos = 0 TO 4
DEBUG ? Valor DIG Todos    ' Muestra los dígitos desde 0 hasta4 de 9742.
NEXT
```

DESPLAZAMIENTO A LA IZQUIERDA (<<)

El operador "*shift left*" (<<), desplaza los bits de un valor, hacia la izquierda, un número especificado de lugares.
Los bits desplazados fuera del borde izquierdo son perdidos, y los bits desplazados adentro por el borde derecho son ceros.

Desplazar hacia la izquierda los bits un valor 'n' número de veces, tiene el mismo efecto que multiplicar ese número por 2 a la n-ésima potencia. Para el caso 100<<3 (desplace los bits del número decimal 100, 3 puestos a la izquierda) es equivalente a 100*2^3. Ejemplo:

En los demás Basic Stamp:

```
Valor     VAR    WORD
Tns       VAR    BYTE
Valor = %1111111111111111
FOR  Tns = 1 TO 16                ' Repite con Tns = 1 a 16.
DEBUG BIN ? Valor << Tns          'desplaza los valores a la izquierda "Tns"
                                        posiciones
NEXT
```

DESPLAZAMIENTO A LA DERECHA (>>)

El operador "*shift right*", desplaza los bits de un valor hacia la derecha un número especificado de veces. Los bits desplazados fuera del borde derecho, son perdidos, y los introducidos por el borde izquierdo son ceros. Desplazando los bits de un valor a la derecha 'n' número de veces tiene el mismo efecto que dividir ese número por 2 a la n-ésima potencia. Para el caso 100>>3 (Desplaza los bits del número decimal 100 a la derecha 3 posiciones) es equivalente a 100/(2^3). Ejemplo:

En los demás Basic Stamp:

```
Valor   VAR    WORD
Tns     VAR   BYTE
Valor = %1111111111111111
FOR Tns = 1 TO 16                 ' Repite con Tns = 1 a 16.
DEBUG BIN ? Valor >> Tns          'Desplaza los valores a la derecha Tns
                                   posiciones.
NEXT
```

REVERSA (REV)

El operador reversa, hace una copia reversada (efecto espejo) de un número especificado de bits, de un valor. Comenzando con el bit mas a la derecha (lsb).
Para el caso %10101101 REV4 haría %1011, una imagen de espejo de los primeros cuatro bits de el valor. Ejemplo:

En los demás Basic Stamp:

```
DEBUG BIN ? %11001011 REV 4                ' Muestra resultado (%1101)
```

AND (&)

El operador &, hace la operación AND entre dos valores. Cada bit de los valores esta sujeto a la lógica digital, es decir:
0 AND 0 = 0
0 AND 1 = 0
1 AND 0 = 0

1 AND 1 = 1
El resultado obtenido por & contendrá 1's solamente en aquellas posiciones en las cuales ambas variables de entrada contengan 1's. Ejemplo:

En el Basic Stamp 1:

```
SYMBOL Valor1 = B0
SYMBOL Valor2 = B1
SYMBOL Resultado = B2
Valor1 = %00001111
Valor2 = %10101101
Resultado = Valor1 & Valor2
DEBUG %Resultado    'Muestra el resultado de la operación AND (%00001101)
```

En los demás Basic Stamp:

```
DEBUG BIN ? %00001111 & %10101101        'Muestra el resultado de la
                                                operación AND (%00001101)
```

OR (|)

El operador (|), hace la operación OR de dos valores. Cada bit del valor esta sujeto a la lógica digital:

0 OR 0 = 0
0 OR 1 = 1
1 OR 0 = 1
1 OR 1 = 1

El resultado obtenido contendrá 1's en las posiciones del bit en la cual una o la otra (o ambas) de las entradas contenga 1's. Ejemplo:

En el Basic Stamp 1:

```
SYMBOL   Valor1 = B0
SYMBOL   Valor2 = B1
SYMBOL   Resultado = B2
Valor1 = %00001111
Valor2 = %10101001
Resultado = Valor1  |  Valor2
DEBUG %Resultado     'Muestra el resultado de la operación OR (%10101111)
```

En los demás Basic Stamp:

```
DEBUG BIN ? %00001111 | %10101001          'Muestra el resultado de la
                                              operación OR (%10101111)
```

XOR (^)

El operador XOR hace la operación XOR entre dos variables. Cada bit de los valores esta sujeto a la lógica digital:

0 XOR 0 = 0
0 XOR 1 = 1
1 XOR 0 = 1
1 XOR 1 = 0

El resultado obtenido contendrá 1's en las posiciones del bit en la cual una o la otra (pero no en ambas) de las entradas contenga 1's. Ejemplo:

En el Basic Stamp 1:

```
SYMBOL   Valor1 = B0
SYMBOL   Valor2 = B1
SYMBOL   Resultado = B2
Valor1 = %00001111
Valor2 = %10101001
Resultado = Valor1 ^ Valor2
DEBUG %Resultado                    'Muestra el resultado de la operación
                                XOR (%10100110)
```

En los demás Basic Stamp:

```
DEBUG BIN ? %00001111 ^ %10101001          'Muestra el resultado de la
                                           operación XOR (%10100110)
```

AND NOT (&/)

Este operador hace la operación AND-NOT entre dos valores, cada bit de los valores esta sujeto a la lógica digital:

```
0 ANDNOT  0 = 0
0 ANDNOT  1 = 0
1 ANDNOT  0 = 1
0 ANDNOT  0 = 0
```

El resultado obtenido contendrá 1's en las posiciones del bit en las cuales el primer valor el 1 y el segundo es cero. Ejemplo:

En el Basic Stamp 1:

```
SYMBOL   Valor1 = B0
SYMBOL   Valor2 = B1
SYMBOL   Resultado = B2
Valor1 = %00001111
Valor2 = %10101001
Resultado = Valor1 &/ Valor2
DEBUG %Resultado                    'Muestra el resultado de la operación
                                AND-NOT (%00000110)
```

OR NOT (| /)

Este operador realiza la operación OR NOT entre dos variables, cada bit de los valores esta sujeto a la lógica digital:

```
0 ORNOT  0 = 1
0 ORNOT  1 = 0
1 ORNOT  0 = 1
1 ORNOT  1 = 1
```

El resultado obtenido contendrá 1's en las posiciones del bit en las cuales el primer valor el 1 o el segundo es cero. Ejemplo:

En el Basic Stamp 1:

```
SYMBOL   Valor1 = B0
SYMBOL   Valor2 = B1
SYMBOL   Resultato = B2
Valor1 = %00001111
Valor2 = %10101001
Resultado = Valor1 |/ Valor2
DEBUG %Resultado                    'Muestra el resultado de la operación OR
                                    NOT (%01011111)
```

XOR NOT (^/)

Este operador realiza la operación XOR NOT entre dos variables, cada bit de los valores esta sujeto a la lógica digital:

```
0 XORNOT  0 = 1
0 XORNOT  1 = 0
1 XORNOT  0 = 0
1 XORNOT  1 = 1
```

El resultado obtenido contendrá 1's en las posiciones del bit en las cuales el primer valor y el segundo valor sean los mismos. Ejemplo:

En el Basic Stamp 1:

```
SYMBOL   Valor1 = B0
SYMBOL   Valor2 = B1
SYMBOL   Resultado = B2
Valor1 = %00001111
Valor2 = %10101001
Resultado = Valor1 ^/ Valor2
DEBUG %Resultado                    'Muestra el resultado de la operación
                                    XOR NOT (%01011001)
```

ANEXO B. Hoja de datos del acelerómetro ADXL202E

ANALOG DEVICES

Low-Cost ±2 *g* Dual-Axis Accelerometer with Duty Cycle Output

ADXL202E*

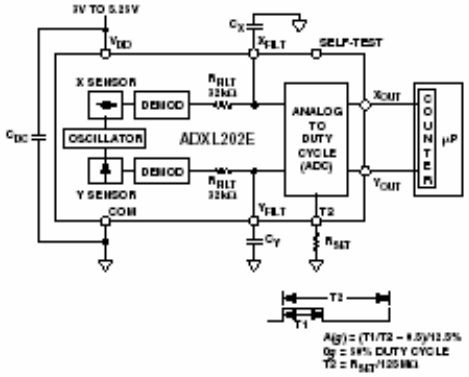
FEATURES

2-Axis Acceleration Sensor on a Single IC Chip
5 mm × 5 mm × 2 mm Ultrasmall Chip Scale Package
2 m*g* Resolution at 60 Hz
Low-Power < 0.6 mA
Direct Interface to Low-Cost Microcontrollers via Duty Cycle Output
BW Adjustment with a Single Capacitor
3 V to 5.25 V Single Supply Operation
1000 *g* Shock Survival

APPLICATIONS

2-Axis Tilt Sensing with Faster Response than Electrolytic, Mercury, or Thermal Sensors
Computer Peripherals
Information Appliances
Alarms and Motion Detectors
Disk Drives
Vehicle Security

FUNCTIONAL BLOCK DIAGRAM



GENERAL DESCRIPTION

The ADXL202E is a low-cost, low-power, complete 2-axis accelerometer with a digital output, all on a single monolithic IC. It is an improved version of the ADXL202AQC/JQC. The ADXL202E will measure accelerations with a full-scale range of ±2 *g*. The ADXL202E can measure both dynamic acceleration (e.g., vibration) and static acceleration (e.g., gravity).

The outputs are analog voltage or digital signals whose duty cycles (ratio of pulsewidth to period) are proportional to acceleration. The duty cycle outputs can be directly measured by a microprocessor counter, without an A/D converter or glue logic. The duty cycle period is adjustable from 0.5 ms to 10 ms via a single resistor (R_{SET}).

The typical noise floor is 200 $\mu g/\sqrt{Hz}$, allowing signals below 2 mg (at 60 Hz bandwidth) to be resolved.

The bandwidth of the accelerometer is set with capacitors C_X and C_Y at the X_{FILT} and Y_{FILT} pins. An analog output can be reconstructed by filtering the duty cycle output.

The ADXL202E is available in 5 mm × 5 mm × 2 mm 8-lead hermetic LCC package.

*Patents Pending

REV. A

One Technology Way, P.O. Box 9106, Norwood, MA 02062-9106, U.S.A.
Tel: 781/329-4700 World Wide Web Site: http://www.analog.com
Fax: 781/326-8703

ADXL202E–SPECIFICATIONS

($T_A = T_{MIN}$ to T_{MAX}, $T_A = 25°C$ for J Grade only, $V_{DD} = 5$ V, $R_{SET} = 125$ kΩ, Acceleration = 0 *g*, unless otherwise noted.)

Parameter	Conditions	TPC[1] Graph	ADXL202JE Min	ADXL202JE Typ	ADXL202JE Max	ADXL202AE Min	ADXL202AE Typ	ADXL202AE Max	Unit
SENSOR INPUT	Each Axis								
Measurement Range[2]			±2			±2			*g*
Nonlinearity	Best Fit Straight Line			0.2			0.2		% of FS
Alignment Error[3]		X		±1			±1		Degrees
Alignment Error	X Sensor to Y Sensor			0.01			0.01		Degrees
Cross-Axis Sensitivity[4]		X		±2			±2		%
SENSITIVITY	Each Axis								
Duty Cycle per *g*	T1/T2, $V_{DD} = 5$ V	X	10.5	12.5	14.5	10	12.5	15	%/*g*
Duty Cycle per *g*	T1/T2, $V_{DD} = 3$ V	X	9.0	11	13.0	8.5	11	13.5	%/*g*
Sensitivity X_{FILT}, Y_{FILT}	$V_{DD} = 5$ V	X	265	312	360	250	312	375	mV/*g*
Sensitivity X_{FILT}, Y_{FILT}	$V_{DD} = 3$ V	X	140	167	195	140	167	200	mV/*g*
Temperature Drift[5]	Delta from 25°C	X		±0.5			±0.5		%
ZERO *g* BIAS LEVEL	Each Axis								
0 *g* Duty Cycle	T1/T2, $V_{DD} = 5$ V	X	34	50	66	30	50	70	%
0 *g* Duty Cycle	T1/T2, $V_{DD} = 3$ V	X	31	50	69	31	50	69	%
0 *g* Voltage X_{FILT}, Y_{FILT}	$V_{DD} = 5$ V	X	2.1	2.5	2.9	2.0	2.5	3.0	V
0 *g* Voltage X_{FILT}, Y_{FILT}	$V_{DD} = 3$ V	X	1.2	1.5	1.8	1.2	1.5	1.8	V
0 *g* Duty Cycle vs. Supply		X		1.0	4.0		1.0	4.0	%/V
0 *g* Offset vs. Temperature[5]	Delta from 25°C	X		2.0			2.0		mg/°C
NOISE PERFORMANCE									
Noise Density	@ 25°C	X		200			200	1000	µg/√Hz rms
FREQUENCY RESPONSE									
3 dB Bandwidth	At Pins X_{FILT}, Y_{FILT}			6			6		kHz
Sensor Resonant Frequency				10			10		kHz
FILTER									
R_{FILT} Tolerance	32 kΩ Nominal			±15			±15		%
Minimum Capacitance	At Pins X_{FILT}, Y_{FILT}		1000			1000			pF
SELF-TEST									
Duty Cycle Change	Self-Test "0" to "1"			10			10		%
DUTY CYCLE OUTPUT STAGE									
F_{SET}	$R_{SET} = 125$ kΩ		0.7		1.3	0.7		1.3	kHz
Output High Voltage	I = 25 µA		V_S – 200 mV			V_S – 200 mV			V
Output Low Voltage	I = 25 µA				200			200	mV
T2 Drift vs. Temperature				50			50		ppm/°C
Rise/Fall Time				200			200		ns
POWER SUPPLY									
Operating Voltage Range			3		5.25	3.0		5.25	V
Quiescent Supply Current				0.6	1.0		0.6	1.0	mA
Turn-On Time	C_{FILT} in µF			160 × C_{FILT} + 0.3			160 × C_{FILT} + 0.3		ms
TEMPERATURE RANGE									
Specified Performance AE						–40		+85	°C
Operating Range			0		70	–40		+85	°C

NOTES

[1]Typical Performance Characteristics.

[2]Guaranteed by measurement of initial offset and sensitivity.

[3]Alignment error is specified as the angle between the true and indicated axis of sensitivity (see TPC 15).

[4]Cross-axis sensitivity is the algebraic sum of the alignment and the inherent sensitivity errors.

[5]Defined as the output change from ambient to maximum temperature or ambient to minimum temperature.

Specifications subject to change without notice.

ABSOLUTE MAXIMUM RATINGS*

Acceleration (Any Axis, Unpowered for 0.5 ms) . . . . . 1000 *g*
Acceleration (Any Axis, Powered for 0.5 ms) . . . . . 500 *g*
$+V_S$ . . . . . −0.3 V to +6.0 V
Output Short Circuit Duration, (Any Pin to Common)
. . . . . Indefinite
Operating Temperature . . . . . −55°C to +125°C
Storage Temperature . . . . . −65°C to +150°C

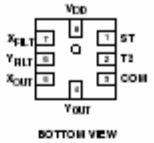
*Stresses above those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only; functional operation of the device at these or any other conditions above those indicate in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

Drops onto hard surfaces can cause shocks of greater than 1000 *g* and exceed the absolute maximum rating of the device. Care should be exercised in handling to avoid damage.

Package Characteristics

Package Weight	θ_{JA}	θ_{JC}	Device
8-Lead LCC	120°C/W	tbd°C/W	<1.0 grams

PIN CONFIGURATION



PIN FUNCTION DESCRIPTIONS

Pin No.	Mnemonic	Description
1	ST	Self-Test
2	T2	Connect R_{SET} to Set T2 Period
3	COM	Common
4	Y_{OUT}	Y-Channel Duty Cycle Output
5	X_{OUT}	X-Channel Duty Cycle Output
6	Y_{FILT}	Y-Channel Filter Pin
7	X_{FILT}	X-Channel Filter Pin
8	V_{DD}	3 V to 5.25 V

ORDERING GUIDE

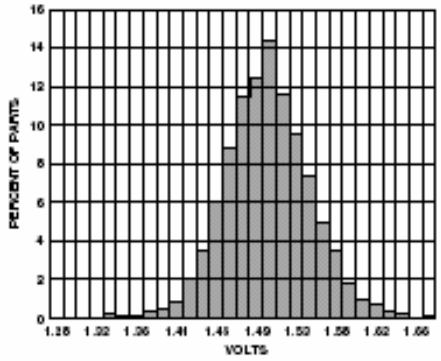
Model	No. of Axes	Specified Voltage	Temperature Range	Package Description	Package Option
ADXL202JE	2	3 V to 5 V	0 to 70°C	8-Lead LCC	E-8
ADXL202AE	2	3 V to 5 V	−40°C to +85°C	8-Lead LCC	E-8

CAUTION

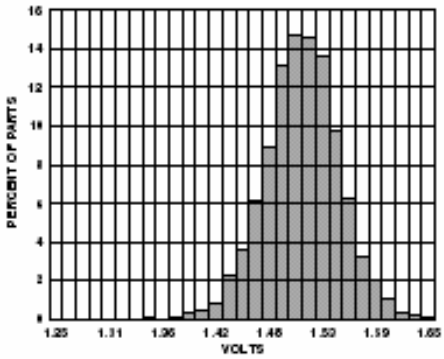
ESD (electrostatic discharge) sensitive device. Electrostatic charges as high as 4000 V readily accumulate on the human body and test equipment and can discharge without detection. Although the ADXL202E features proprietary ESD protection circuitry, permanent damage may occur on devices subjected to high-energy electrostatic discharges. Therefore, proper ESD precautions are recommended to avoid performance degradation or loss of functionality.

ADXL202E–Typical Performance Characteristics*

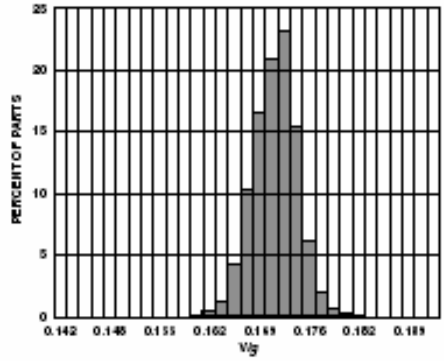
V_{DD} = 3 V



TPC 1. X-Axis Zero g Bias Distribution at X_{FILT}, V_{DD} = 3 V

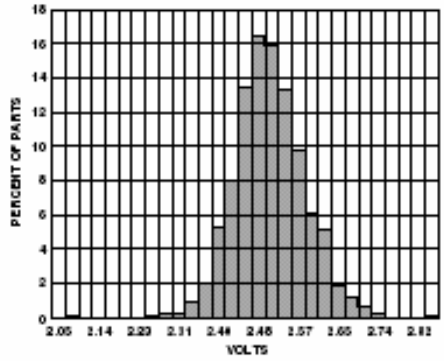


TPC 2. Y-Axis Zero g Bias Distribution at Y_{FILT}, V_{DD} = 3 V

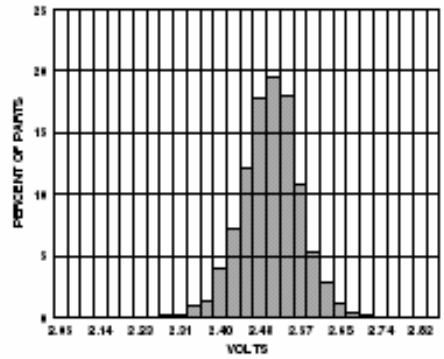


TPC 3. X-Axis Sensitivity Distribution at X_{FILT}, V_{DD} = 3 V

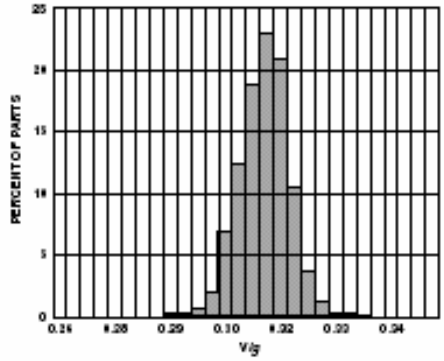
V_{DD} = 5 V



TPC 4. X-Axis Zero g Bias Distribution at X_{FILT}, V_{DD} = 5 V



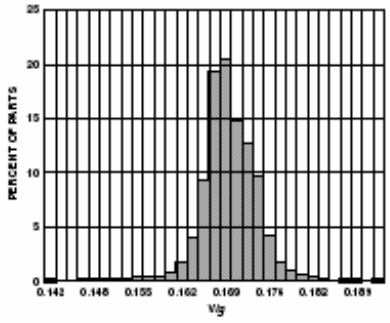
TPC 5. Y-Axis Zero g Bias Distribution at Y_{FILT}, V_{DD} = 5 V



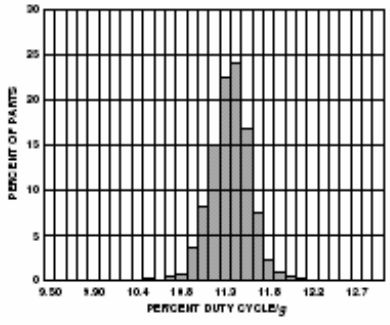
TPC 6. X-Axis Sensitivity Distribution at X_{FILT}, V_{DD} = 5 V

*Data taken from 4500 parts over 3 lots minimum.

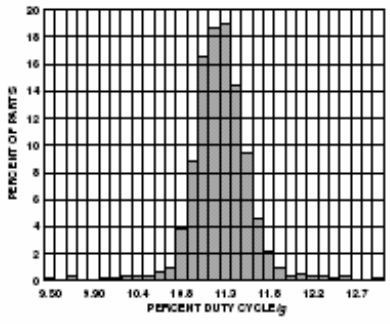
V_{DD} = 3 V



TPC 7. Y-Axis Sensitivity Distribution at Y_{FILT}, V_{DD} = 3 V

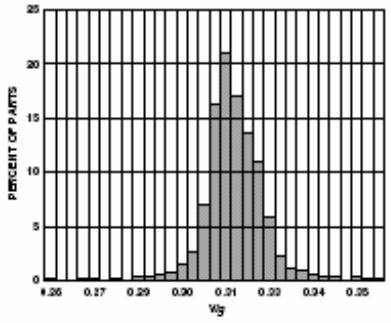


TPC 8. X-Axis Sensitivity at X_{OUT}, V_{DD} = 3 V

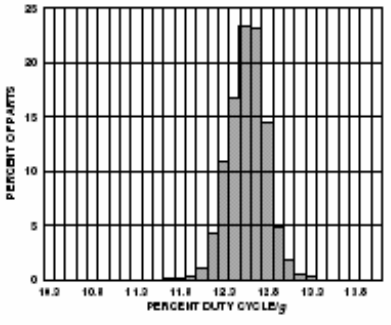


TPC 9. Y-Axis Sensitivity at Y_{OUT}, V_{DD} = 3 V

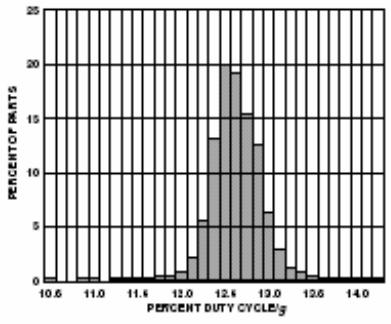
V_{DD} = 5 V



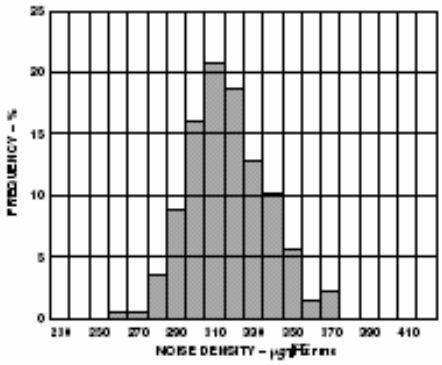
TPC 10. Y-Axis Sensitivity Distribution at Y_{FILT}, V_{DD} = 5 V



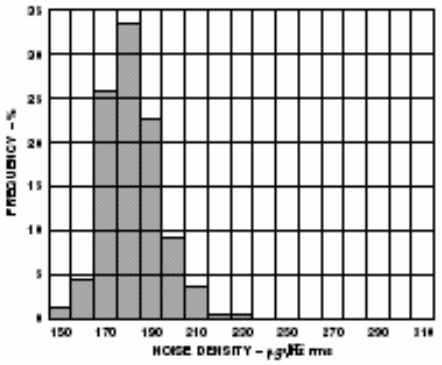
TPC 11. X-Axis Sensitivity at X_{OUT}, V_{DD} = 5 V



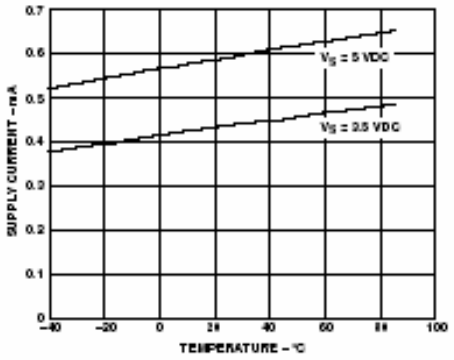
TPC 12. Y-Axis Sensitivity at Y_{OUT}, V_{DD} = 5 V



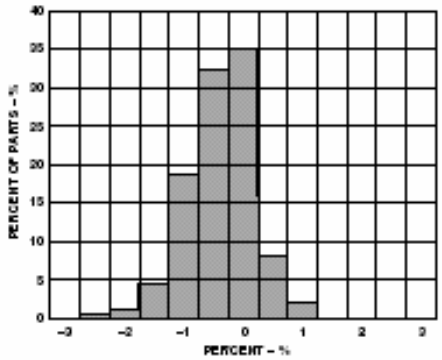
TPC 13. Noise Density Distribution, V_{DD} = 3 V



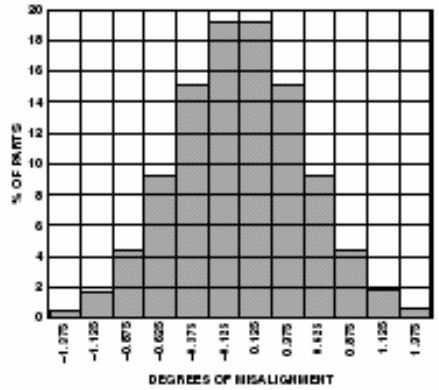
TPC 16. Noise Density Distribution, V_{DD} = 5 V



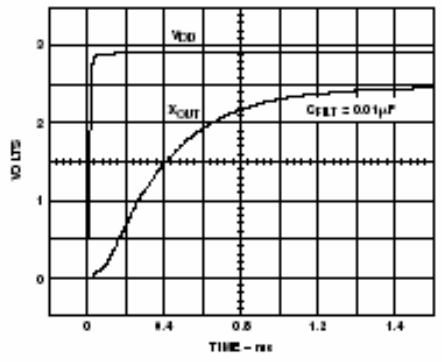
TPC 14. Typical Supply Current vs. Temperature



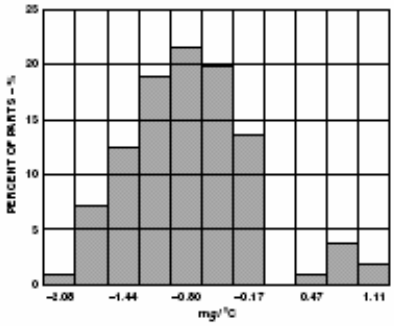
TPC 17. Cross-Axis Sensitivity Distribution



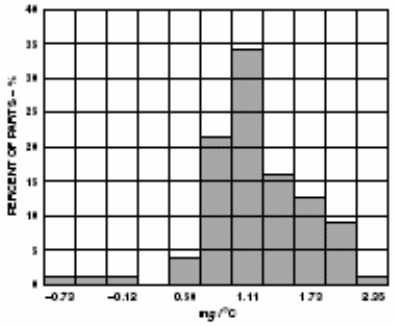
TPC 15. Rotational Die Alignment



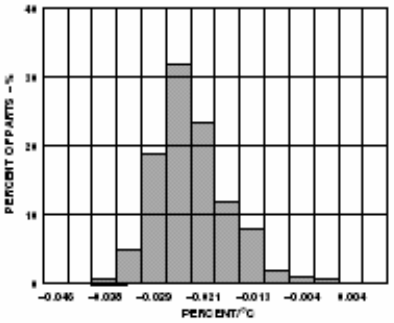
TPC 18. Typical Turn-On Time



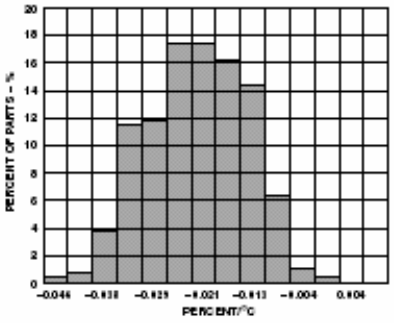
TPC 19. X-Axis Zero g Drift Due to Temperature Distribution, –40°C to +85°C



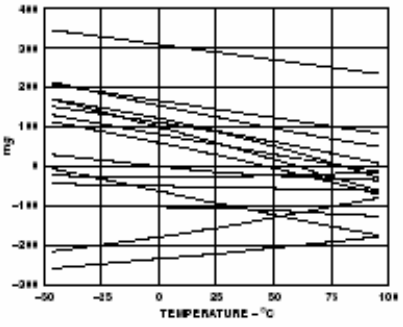
TPC 22. Y-Axis Zero g Drift Due to Temperature Distribution, –40°C to +85°C



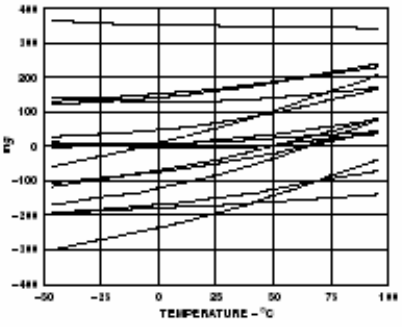
TPC 20. X-Axis Sensitivity Drift at X_{FILT} Due to Temperature Distribution, –40°C to +85°C



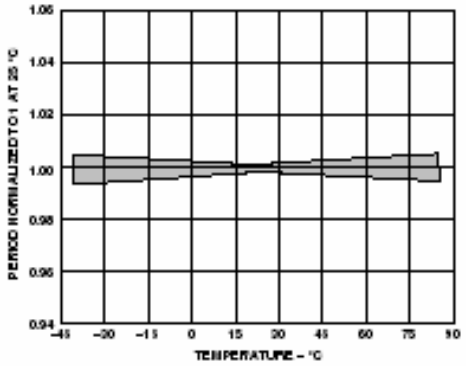
TPC 23. Y-Axis Sensitivity Drift at Y_{FILT} Due to Temperature Distribution, –40°C to +85°C



TPC 21. Typical X-Axis Zero g vs. Output for 16 Parts



TPC 24. Typical Y-Axis Zero g vs. Output for 16 Parts



TPC 25. Normalized DCM Period (T2) vs. Temperature

DEFINITIONS

T1	Length of the "on" portion of the cycle.
T2	Length of the total cycle.
Duty Cycle	Ratio of the "on" time (T1) of the cycle to the total cycle (T2). Defined as T1/T2 for the ADXL202E/ ADXL210.
Pulsewidth	Time period of the "on" pulse. Defined as T1 for the ADXL202E/ADXL210.

THEORY OF OPERATION

The ADXL202E is a complete, dual-axis acceleration measurement system on a single monolithic IC. It contains a polysilicon surface-micromachined sensor and signal conditioning circuitry to implement an open loop acceleration measurement architecture. For each axis, an output circuit converts the analog signal to a duty cycle modulated (DCM) digital signal that can be decoded with a counter/timer port on a microprocessor. The ADXL202E is capable of measuring both positive and negative accelerations to at least ±2 *g*. The accelerometer can measure static acceleration forces such as gravity, allowing it to be used as a tilt sensor.

The sensor is a surface micromachined polysilicon structure built on top of the silicon wafer. Polysilicon springs suspend the structure over the surface of the wafer and provide a resistance against acceleration forces. Deflection of the structure is measured using a differential capacitor that consists of independent fixed plates and central plates attached to the moving mass. The fixed plates are driven by 180° out of phase square waves. An acceleration will deflect the beam and unbalance the differential capacitor, resulting in an output square wave whose amplitude is proportional to acceleration. Phase sensitive demodulation techniques are then used to rectify the signal and determine the direction of the acceleration.

The output of the demodulator drives a duty cycle modulator (DCM) stage through a 32 kΩ resistor. At this point a pin is available on each channel to allow the user to set the signal bandwidth of the device by adding a capacitor. This filtering improves measurement resolution and helps prevent aliasing.

After being low-pass filtered, the analog signal is converted to a duty cycle modulated signal by the DCM stage. A single resistor sets the period for a complete cycle (T2), which can be set between 0.5 ms and 10 ms (see Figure 12). A 0 *g* acceleration produces a nominally 50% duty cycle. The acceleration signal can be determined by measuring the length of the T1 and T2 pulses with a counter/timer or with a polling loop using a low cost microcontroller.

An analog output voltage can be obtained either by buffering the signal from the X_{FILT} and Y_{FILT} pin, or by passing the duty cycle signal through an RC filter to reconstruct the dc value.

The ADXL202E will operate with supply voltages as low as 3.0 V or as high as 5.25 V.

Figure 1. Typical Output Duty Cycle

APPLICATIONS

POWER SUPPLY DECOUPLING

For most applications a single 0.1 µF capacitor, C_{DC}, will adequately decouple the accelerometer from signal and noise on the power supply. However, in some cases, especially where digital devices such as microcontrollers share the same power supply, digital noise on the supply may cause interference on the ADXL202E output. This may be observed as a slowly undulating fluctuation of voltage at X_{FILT} and Y_{FILT}. If additional decoupling is needed, a 100 Ω (or smaller) resistor or ferrite beads, may be inserted in the supply line of the ADXL202E.

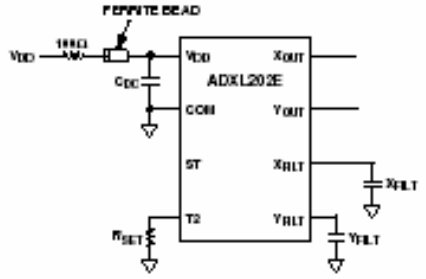


Figure 2.

DESIGN PROCEDURE FOR THE ADXL202E

The design procedure for using the ADXL202E with a duty cycle output involves selecting a duty cycle period and a filter capacitor. A proper design will take into account the application requirements for bandwidth, signal resolution and acquisition time, as discussed in the following sections.

Decoupling Capacitor C_{DC}

A 0.1 µF capacitor is recommended from V_{DD} to COM for power supply decoupling.

ST

The ST pin controls the self-test feature. When this pin is set to V_{DD}, an electrostatic force is exerted on the beam of the accelerometer. The resulting movement of the beam allows the user to test if the accelerometer is functional. The typical change in output will be 10% at the duty cycle outputs (corresponding to 800 mg). This pin may be left open circuit or connected to common in normal use.

Duty Cycle Decoding

The ADXL202E's digital output is a duty cycle modulator. Acceleration is proportional to the ratio T1/T2. The nominal output of the ADXL202E is:

$$0\ g = 50\%\ Duty\ Cycle$$

Scale factor is 12.5% Duty Cycle Change per *g*

These nominal values are affected by the initial tolerance of the device including zero *g* offset error and sensitivity error.

T2 does not have to be measured for every measurement cycle. It need only be updated to account for changes due to temperature, (a relatively slow process). Since the T2 time period is shared by both X and Y channels, it is necessary only to measure it on one channel of the ADXL202E. Decoding algorithms for various microcontrollers have been developed. Consult the appropriate Application Note.

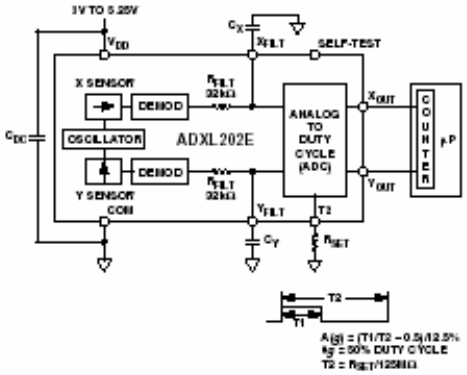


Figure 3. Block Diagram

Setting the Bandwidth Using C_X and C_Y

The ADXL202E has provisions for bandlimiting the X_{FILT} and Y_{FILT} pins. Capacitors must be added at these pins to implement low-pass filtering for antialiasing and noise reduction. The equation for the 3 dB bandwidth is:

$$F_{-3\,dB} = \frac{1}{\left(2\,\pi\,(32\ k\Omega) \times C(x,y)\right)}$$

or, more simply, $F_{-3dB} = \frac{5\,\mu F}{C_{(X,Y)}}$

The tolerance of the internal resistor (R_{FILT}), can vary typically as much as ±15% of its nominal value of 32 kΩ; so the bandwidth will vary accordingly. A minimum capacitance of 1000 pF for $C_{(X,Y)}$ is required in all cases.

Table I. Filter Capacitor Selection, C_X and C_Y

Bandwidth	Capacitor Value
10 Hz	0.47 µF
50 Hz	0.10 µF
100 Hz	0.05 µF
200 Hz	0.027 µF
500 Hz	0.01 µF
5 kHz	0.001 µF

Setting the DCM Period with R_{SET}

The period of the DCM output is set for both channels by a single resistor from R_{SET} to ground. The equation for the period is:

$$T2 = \frac{R_{SET}\ (\Omega)}{125\ M\Omega}$$

A 125 kΩ resistor will set the duty cycle repetition rate to approximately 1 kHz, or 1 ms. The device is designed to operate at duty cycle periods between 0.5 ms and 10 ms.

Table II. Resistor Values to Set T2

T2	R_{SET}
1 ms	125 kΩ
2 ms	250 kΩ
5 ms	625 kΩ
10 ms	1.25 MΩ

Note that the R_{SET} should always be included, even if only an analog output is desired. Use an R_{SET} value between 500 kΩ and 2 MΩ when taking the output from X_{FILT} or Y_{FILT}. The R_{SET} resistor should be place close to the T2 Pin to minimize parasitic capacitance at this node.

Selecting the Right Accelerometer

For most tilt sensing applications the ADXL202E is the most appropriate accelerometer. Its higher sensitivity (12.5%/*g*) allows the user to use a lower speed counter for PWM decoding while maintaining high resolution. The ADXL210 should be used in applications where accelerations of greater than ±2 *g* are expected.

MICROCOMPUTER INTERFACES

The ADXL202E is specifically designed to work with low-cost microcontrollers. Specific code sets, reference designs, and application notes are available from the factory. This section will outline a general design procedure and discuss the various trade-offs that need to be considered.

The designer should have some idea of the required performance of the system in terms of:

Resolution: the smallest signal change that needs to be detected.

Bandwidth: the highest frequency that needs to be detected.

Acquisition Time: the time that will be available to acquire the signal on each axis.

These requirements will help to determine the accelerometer bandwidth, the speed of the microcontroller clock and the length of the T2 period.

When selecting a microcontroller it is helpful to have a counter timer port available. The microcontroller should have provisions for software calibration. While the ADXL202E is a highly accurate accelerometer, it has a wide tolerance for initial offset. The easiest way to null this offset is with a calibration factor saved on the microcontroller or by a user calibration for zero *g*. In the case where the offset is calibrated during manufacture, there are several options, including external EEPROM and microcontrollers with "one-time programmable" features.

DESIGN TRADE-OFFS FOR SELECTING FILTER CHARACTERISTICS: THE NOISE/BW TRADE-OFF

The accelerometer bandwidth selected will determine the measurement resolution (smallest detectable acceleration). Filtering can be used to lower the noise floor and improve the resolution of the accelerometer. Resolution is dependent on both the analog filter bandwidth at X_{FILT} and Y_{FILT} and on the speed of the microcontroller counter.

The analog output of the ADXL202E has a typical bandwidth of 5 kHz, while the duty cycle modulators' bandwidth is 500 Hz. The user must filter the signal at this point to limit aliasing errors. To minimize DCM errors the analog bandwidth should be less than 1/10 the DCM frequency. Analog bandwidth may be increased to up to 1/2 the DCM frequency in many applications. This will result in greater dynamic error generated at the DCM.

The analog bandwidth may be further decreased to reduce noise and improve resolution. The ADXL202E noise has the characteristics of white Gaussian noise that contributes equally at all frequencies and is described in terms of µg per root Hz; i.e., the noise is proportional to the square root of the bandwidth of the accelerometer. It is recommended that the user limit bandwidth to the lowest frequency needed by the application, to maximize the resolution and dynamic range of the accelerometer.

With the single pole roll-off characteristic, the typical noise of the ADXL202E is determined by the following equation:

$$Noise\ (rms) = \left(200\ \mu g/\sqrt{Hz}\right) \times \left(\sqrt{BW \times 1.6}\right)$$

At 100 Hz the noise will be:

$$Noise\ (rms) = \left(200\ \mu g/\sqrt{Hz}\right) \times \left(\sqrt{100 \times (1.6)}\right) = 2.53\ mg$$

Often the peak value of the noise is desired. Peak-to-peak noise can only be estimated by statistical methods. Table III is useful for estimating the probabilities of exceeding various peak values, given the rms value.

Table III. Estimation of Peak-to-Peak Noise

Nominal Peak-to-Peak Value	% of Time that Noise Will Exceed Nominal Peak-to-Peak Value
2.0 × rms	32%
4.0 × rms	4.6%
6.0 × rms	0.27%
8.0 × rms	0.006%

The peak-to-peak noise value will give the best estimate of the uncertainty in a single measurement.

Table IV gives typical noise output of the ADXL202E for various C_X and C_Y values.

Table IV. Filter Capacitor Selection, C_X and C_Y

Bandwidth	C_X, C_Y	rms Noise	Peak-to-Peak Noise Estimate 95% Probability (rms × 4)
10 Hz	0.47 µF	0.8 mg	3.2 mg
50 Hz	0.10 µF	1.8 mg	7.2 mg
100 Hz	0.05 µF	2.5 mg	10.1 mg
200 Hz	0.027 µF	3.6 mg	14.3 mg
500 Hz	0.01 µF	5.7 mg	22.6 mg

CHOOSING T2 AND COUNTER FREQUENCY: DESIGN TRADE-OFFS

The noise level is one determinant of accelerometer resolution. The second relates to the measurement resolution of the counter when decoding the duty cycle output.

The ADXL202E's duty cycle converter has a resolution of approximately 14 bits; better resolution than the accelerometer itself. The actual resolution of the acceleration signal is, however, limited by the time resolution of the counting devices used to decode the duty cycle. The faster the counter clock, the higher the resolution of the duty cycle and the shorter the T2 period can be for a given resolution. The following table shows some of the trade-offs. It is important to note that this is the resolution due to the microprocessors' counter. It is probable that the accelerometer's noise floor may set the lower limit on the resolution, as discussed in the previous section.

Table V. Trade-Offs Between Microcontroller Counter Rate, T2 Period, and Resolution of Duty Cycle Modulator

T2 (ms)	R_{SET} (kΩ)	ADXL202E Sample Rate	Counter-Clock Rate (MHz)	Counts per T2 Cycle	Counts per g	Resolution (mg)
1.0	124	1000	2.0	2000	250	4.0
1.0	124	1000	1.0	1000	125	8.0
1.0	124	1000	0.5	500	62.5	16.0
5.0	625	200	2.0	10000	1250	0.8
5.0	625	200	1.0	5000	625	1.6
5.0	625	200	0.5	2500	312.5	3.2
10.0	1250	100	2.0	20000	2500	0.4
10.0	1250	100	1.0	10000	1250	0.8
10.0	1250	100	0.5	5000	625	1.6

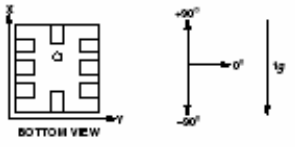
STRATEGIES FOR USING THE DUTY CYCLE OUTPUT WITH MICROCONTROLLERS

Application notes outlining various strategies for using the duty cycle output with low cost microcontrollers are available from the factory.

USING THE ADXL202E AS A DUAL-AXIS TILT SENSOR

One of the most popular applications of the ADXL202E is tilt measurement. An accelerometer uses the force of gravity as an input vector to determine orientation of an object in space.

An accelerometer is most sensitive to tilt when its sensitive axis is perpendicular to the force of gravity, i.e., parallel to the earth's surface. At this orientation its sensitivity to changes in tilt is highest. When the accelerometer is oriented on axis to gravity, i.e., near its +1 *g* or −1 *g* reading, the change in output acceleration per degree of tilt is negligible. When the accelerometer is perpendicular to gravity, its output will change nearly 17.5 mg per degree of tilt, but at 45° degrees it is changing only at 12.2 mg per degree and resolution declines. The following table illustrates the changes in the X and Y axes as the device is tilted ±90° through gravity.



X Axis Orientation to Horizon (°)	X Output (g)	X Output: Δ per Degree of Tilt (mg)	Y Output (g)	Y Output: Δ per Degree of Tilt (mg)
–90	–1.000	–0.2	0.000	17.5
–75	–0.966	4.4	0.259	16.9
–60	–0.866	8.6	0.500	15.2
–45	–0.707	12.2	0.707	12.4
–30	–0.500	15.2	0.866	8.9
–15	–0.259	16.8	0.966	4.7
0	0.000	17.5	1.000	0.2
15	0.259	16.9	0.966	–4.4
30	0.500	15.2	0.866	–8.6
45	0.707	12.4	0.707	–12.2
60	0.866	8.9	0.500	–15.0
75	0.966	4.7	0.259	–16.8
90	1.000	0.2	0.000	–17.5

Figure 4. How the X and Y Axes Respond to Changes in Tilt

A DUAL AXIS TILT SENSOR: CONVERTING ACCELERATION TO TILT

When the accelerometer is oriented so both its X and Y axes are parallel to the earth's surface it can be used as a two axis tilt sensor with a roll and a pitch axis. Once the output signal from the accelerometer has been converted to an acceleration that varies between −1 *g* and +1 *g*, the output tilt in degrees is calculated as follows:

$$Pitch = ASIN\ (Ax/1\ g)$$
$$Roll = ASIN\ (Ay/1\ g)$$

Be sure to account for overranges. It is possible for the accelerometers to output a signal greater than ±1 *g* due to vibration, shock or other accelerations.

MEASURING 360° OF TILT

It is possible to measure a full 360° of orientation through gravity by using two accelerometers oriented perpendicular to one another (see Figure 5). When one sensor is reading a maximum change in output per degree, the other is at its minimum.

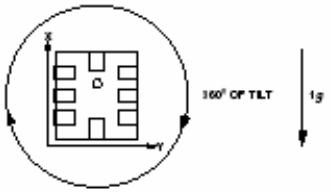


Figure 5. Using a Two-Axis Accelerometer to Measure 360° of Tilt

USING THE ANALOG OUTPUT

The ADXL202E was specifically designed for use with its digital outputs, but has provisions to provide analog outputs as well.

Duty Cycle Filtering

An analog output can be reconstructed by filtering the duty cycle output. This technique requires only passive components. The duty cycle period (T2) should be set to <1 ms. An RC filter with a 3 dB point at least a factor of >10 less than the duty cycle frequency is connected to the duty cycle output. The filter resistor should be no less than 100 kΩ to prevent loading of the output stage. The analog output signal will be ratiometric to the supply voltage. The advantage of this method is an output scale factor of approximately double the analog output. Its disadvantage is that the frequency response will be lower than when using the X_{FILT}, Y_{FILT} output.

X_{FILT}, Y_{FILT} Output

The second method is to use the analog output present at the X_{FILT} and Y_{FILT} pin. Unfortunately, these pins have a 32 kΩ output impedance and are not designed to drive a load directly. An op amp follower may be required to buffer this pin. The advantage of this method is that the full 5 kHz bandwidth of the accelerometer is available to the user. A capacitor still must be added at this point for filtering. The duty cycle converter should be kept running by using R_{SET} <10 MΩ. Note that the accelerometer offset and sensitivity are ratiometric to the supply voltage. The offset and sensitivity are nominally:

0 *g* Offset = $V_{DD}/2$

ADXL202E Sensitivity = (60 mV × V_S)/*g*

USING THE ADXL202E IN VERY LOW POWER APPLICATIONS

An application note outlining low power strategies for the ADXL202E is available. Some key points are presented here. It is possible to reduce the ADXL202E's average current from 0.6 mA to less than 20 µA by using the following techniques:

1. Power Cycle the accelerometer.
2. Run the accelerometer at a Lower Voltage, (Down to 3 V).

Power Cycling with an External A/D

Depending on the value of the X_{FILT} capacitor, the ADXL202E is capable of turning on and giving a good reading in 1.6 ms. Most microcontroller based A/Ds can acquire a reading in another 25 µs. Thus it is possible to turn on the ADXL202E and take a reading in <2 ms. If we assume that a 20 Hz sample rate is sufficient, the total current required to take 20 samples is 2 ms × 20 samples/s × 0.6 mA = 24 µA average current. Running the part at 3 V will reduce the supply current from 0.6 mA to 0.4 mA, bringing the average current down to 16 µA.

The A/D should read the analog output of the ADXL202E at the X_{FILT} and Y_{FILT} pins. A buffer amplifier is recommended, and may be required in any case to amplify the analog output to give enough resolution with an 8-bit to 10-bit converter.

Power Cycling When Using the Digital Output

An alternative is to run the microcontroller at a higher clock rate and put it into shutdown between readings, allowing the use of the digital output. In this approach the ADXL202E should be set at its fastest sample rate (T2 = 0.5 ms), with a 500 Hz filter at X_{FILT} and Y_{FILT}. The concept is to acquire a reading as quickly as possible and then shut down the ADXL202E and the microcontroller until the next sample is needed.

In either of the above approaches, the ADXL202E can be turned on and off directly using a digital port pin on the microcontroller to power the accelerometer without additional components.

CALIBRATING THE ADXL202E/ADXL210

The initial value of the offset and scale factor for the ADXL202E will require calibration for applications such as tilt measurement. The ADXL202E architecture has been designed so that these calibrations take place in the software of the microcontroller used to decode the duty cycle signal. Calibration factors can be stored in EEPROM or determined at turn-on and saved in dynamic memory.

For low g applications, the force of gravity is the most stable, accurate and convenient acceleration reference available. A reading of the 0 g point can be determined by orientating the device parallel to the earth's surface and then reading the output.

A more accurate calibration method is to make measurements at +1 g and −1 g. The sensitivity can be determined by the two measurements.

To calibrate, the accelerometer's measurement axis is pointed directly at the earth. The 1 g reading is saved and the sensor is turned 180° to measure −1 g. Using the two readings, the sensitivity is:

Let A = Accelerometer output with axis oriented to +1 g
Let B = Accelerometer output with axis oriented to −1 g then:

$$Sensitivity = [A - B]/2\ g$$

For example, if the +1 g reading (A) is 55% duty cycle and the −1 g reading (B) is 32% duty cycle, then:

$$Sensitivity = [55\% - 32\%]/2\ g = 11.5\%/g$$

These equations apply whether the output is analog or duty cycle.

Application notes outlining algorithms for calculating acceleration from duty cycle and automated calibration routines are available from the factory.

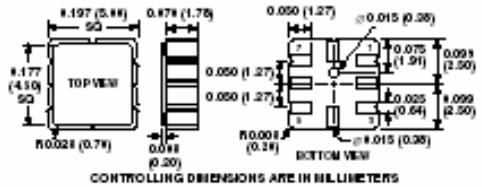
C02064–2.5–10/00 (rev. A)

PRINTED IN U.S.A.

OUTLINE DIMENSIONS

Dimensions shown in inches and (mm).

8-Terminal Ceramic Leadless Chip Carrier (E-8)



ANEXO C. Hoja de datos memoria XICOR X25128P

APPLICATION NOTE
AVAILABLE
AN61

128K **X25128** 16K x 8 Bit

SPI Serial E²PROM with Block Lock™ Protection

FEATURES

- 2MHz Clock Rate
- SPI Modes (0,0 & 1,1)
- 16K X 8 Bits
  - —32 Byte Page Mode
- Low Power CMOS
  - —<1µA Standby Current
  - —<5mA Active Current
- 2.7V To 5.5V Power Supply
- Block Lock Protection
  - —Protect 1/4, 1/2 or all of E²PROM Array
- Built-in Inadvertent Write Protection
  - —Power-Up/Power-Down protection circuitry
  - —Write Enable Latch
  - —Write Protect Pin
- Self-Timed Write Cycle
  - —5ms Write Cycle Time (Typical)
- High Reliability
  - —Endurance: 1 Million cycles
  - —Data Retention: 100 Years
  - —ESD protection: 2000V on all pins
- Packages
  - —8-Lead XBGA
  - —8, 14-Lead SOIC
  - —8-Lead PDIP
  - —8-Lead TSSOP

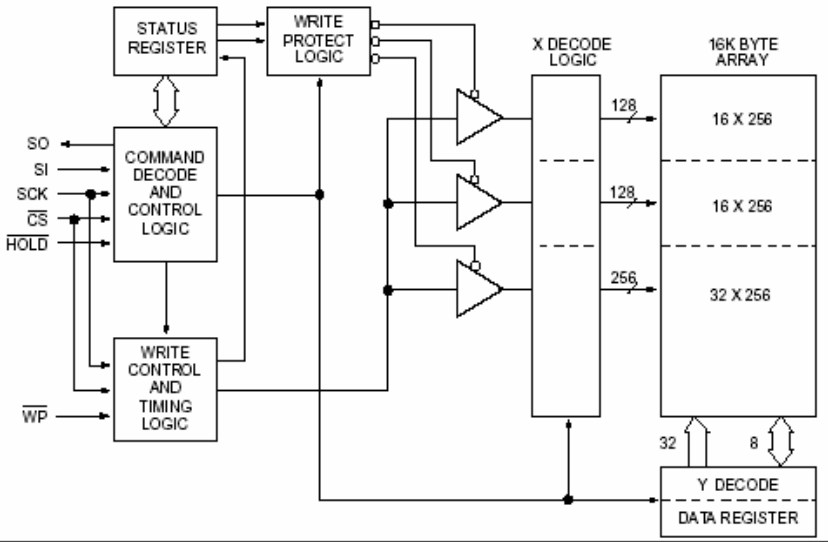
DESCRIPTION

The X25128 is a CMOS 131,072-bit serial E²PROM, internally organized as 16K x 8. The X25128 features a Serial Peripheral Interface (SPI) and software protocol allowing operation on a simple three-wire bus. The bus signals are a clock input (SCK) plus separate data in (SI) and data out (SO) lines. Access to the device is controlled through a chip select ($\overline{CS}$) input, allowing any number of devices to share the same bus.

The X25128 also features two additional inputs that provide the end user with added flexibility. By asserting the $\overline{HOLD}$ input, the X25128 will ignore transitions on its inputs, thus allowing the host to service higher priority interrupts. The $\overline{WP}$ input can be used as a hardwire input to the X25128 disabling all write attempts to the status register, thus providing a mechanism for limiting end user capability of altering 0, 1/4, 1/2 or all of the memory.

The X25128 utilizes Xicor's proprietary Direct Write™ cell, providing a minimum endurance of 100,000 cycles and a minimum data retention of 100 years.

FUNCTIONAL DIAGRAM



PIN DESCRIPTIONS

Serial Output (SO)

SO is a push/pull serial data output pin. During a read cycle, data is shifted out on this pin. Data is clocked out by the falling edge of the serial clock.

Serial Input (SI)

SI is the serial data input pin. All opcodes, byte addresses, and data to be written to the memory are input on this pin. Data is latched by the rising edge of the serial clock.

Serial Clock (SCK)

The Serial Clock controls the serial bus timing for data input and output. Opcodes, addresses, or data present on the SI pin are latched on the rising edge of the clock input, while data on the SO pin change after the falling edge of the clock input.

Chip Select ($\overline{CS}$)

When $\overline{CS}$ is high, the X25128 is deselected and the SO output pin is at high impedance and unless an internal write operation is underway, the X25128 will be in the standby power mode. $\overline{CS}$ low enables the X25128, placing it in the active power mode. It should be noted that after power-up, a high to low transition on $\overline{CS}$ is required prior to the start of any operation.

Write Protect ($\overline{WP}$)

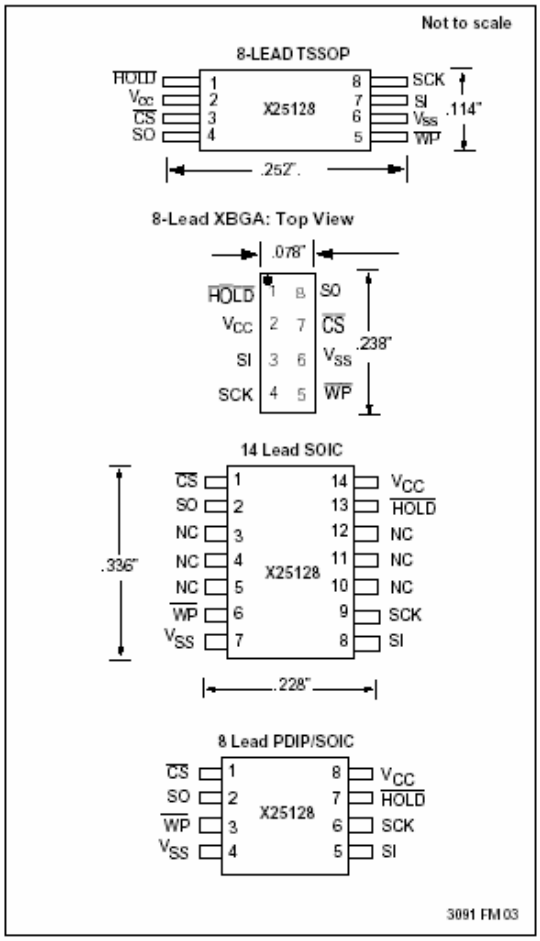
When $\overline{WP}$ is low and the nonvolatile bit WPEN is "1", nonvolatile writes to the X25128 status register are disabled, but the part otherwise functions normally. When $\overline{WP}$ is held high, all functions, including nonvolatile writes operate normally. $\overline{WP}$ going low while $\overline{CS}$ is still low will interrupt a write to the X25128 status register. If the internal write cycle has already been initiated, $\overline{WP}$ going low will have no effect on a write.

The $\overline{WP}$ pin function is blocked when the WPEN bit in the status register is "0". This allows the user to install the X25128 in a system with $\overline{WP}$ pin grounded and still be able to write to the status register. The $\overline{WP}$ pin functions will be enabled when the WPEN bit is set "1".

Hold ($\overline{HOLD}$)

$\overline{HOLD}$ is used in conjunction with the $\overline{CS}$ pin to select the device. Once the part is selected and a serial sequence is underway, $\overline{HOLD}$ may be used to pause the serial communication with the controller without resetting the serial sequence. To pause, $\overline{HOLD}$ must be brought low while SCK is Low. To resume communication, $\overline{HOLD}$ is brought high, again while SCK is low. If the pause feature is not used, $\overline{HOLD}$ should be held high at all times.

PIN CONFIGURATION



PIN NAMES

Symbol	Description
$\overline{CS}$	Chip Select Input
SO	Serial Output
SI	Serial Input
SCK	Serial Clock Input
$\overline{WP}$	Write Protect Input
V_{SS}	Ground
V_{CC}	Supply Voltage
$\overline{HOLD}$	Hold Input
NC	No Connect

3091 FM T01

PRINCIPLES OF OPERATION

The X25128 is a 8K x 8 E^2PROM designed to interface directly with the synchronous serial peripheral interface (SPI) of many popular microcontroller families.

The X25128 contains an 8-bit instruction register. It is accessed via the SI input, with data being clocked in on the rising SCK. $\overline{CS}$ must be low and the $\overline{HOLD}$ and $\overline{WP}$ inputs must be high during the entire operation. The $\overline{WP}$ input is "Don't Care" if WPEN is set "0".

Table 1 contains a list of the instructions and their opcodes. All instructions, addresses and data are transferred MSB first.

Data input is sampled on the first rising edge of SCK after $\overline{CS}$ goes low. SCK is static, allowing the user to stop the clock and then resume operations. If the clock line is shared with other peripheral devices on the SPI bus, the user can assert the $\overline{HOLD}$ input to place the X25128 into a "PAUSE" condition. After releasing $\overline{HOLD}$, the X25128 will resume operation from the point when $\overline{HOLD}$ was first asserted.

Write Enable Latch

The X25128 contains a "write enable" latch. This latch must be SET before a write operation will be completed internally. The WREN instruction will set the latch and the WRDI instruction will reset the latch. This latch is automatically reset upon a power-on condition and after the completion of a byte, page, or status register write cycle.

Status Register

The RDSR instruction provides access to the status register. The status register may be read at any time, even during a write cycle. The status register is formatted as follows:

7	6	5	4	3	2	1	0
WPEN	X	X	X	BP1	BP0	WEL	WIP

3091 FM T02

WPEN, BP0 and BP1 are set by the WRSR instruction. WEL and WIP are read-only and automatically set by other operations.

The Write-In-Process (WIP) bit indicates whether the X25128 is busy with a write operation. When set to a "1", a write is in progress, when set to a "0", no write is in progress. During a write, all other bits are set to "1".

The Write Enable Latch (WEL) bit indicates the status of the "write enable" latch. When set to a "1", the latch is set, when set to a "0", the latch is reset.

The Block Protect (BP0 and BP1) bits are nonvolatile and allows the user to select one of four levels of protection. The X25128 is divided into four 32,768-bit segments. One, two, or all four of the segments may be protected. That is, the user may read the segments but will be unable to alter (write) data within the selected segments. The partitioning is controlled as illustrated below.

Status Register Bits		Array Addresses Protected
BP1	BP0	
0	0	None
0	1	$3000–$3FFF
1	0	$2000–$3FFF
1	1	$0000–$3FFF

3091 PGM T03

Table 1. Instruction Set

Instruction Name	Instruction Format*	Operation
WREN	0000 0110	Set the Write Enable Latch (Enable Write Operations)
WRDI	0000 0100	Reset the Write Enable Latch (Disable Write Operations)
RDSR	0000 0101	Read Status Register
WRSR	0000 0001	Write Status Register
READ	0000 0011	Read Data from Memory Array beginning at selected address
WRITE	0000 0010	Write Data to Memory Array beginning at Selected Address (1 to 32 Bytes)

3091 PGM T04

*Instructions are shown MSB in leftmost position. Instructions are transferred MSB first.

Write-Protect Enable

The Write-Protect-Enable (WPEN) is available for the X25128 as a nonvolatile enable bit for the $\overline{WP}$ pin.

WPEN	$\overline{WP}$	WEL	Protected Blocks	Unprotected Blocks	Status Register
0	X	0	Protected	Protected	Protected
0	X	1	Protected	Writable	Writable
1	Low	0	Protected	Protected	Protected
1	Low	1	Protected	Writable	Protected
X	High	0	Protected	Protected	Protected
X	High	1	Protected	Writable	Writable

3091 PGM T0s.1

The Write Protect ($\overline{WP}$) pin and the nonvolatile Write Protect Enable (WPEN) bit in the Status Register control the programmable hardware write protect feature. Hardware write protection is enabled when $\overline{WP}$ pin is low, and the WPEN bit is "1". Hardware write protection is disabled when either the $\overline{WP}$ pin is high or the WPEN bit is "0". When the chip is hardware write protected, nonvolatile writes are disabled to the Status Register, including the Block Protect bits and the WPEN bit itself, as well as the block-protected sections in the memory array. Only the sections of the memory array that are not block-protected can be written.

Note: Since the WPEN bit is write protected, it cannot be changed back to a "0", as long as the $\overline{WP}$ pin is held low.

Clock and Data Timing

Data input on the SI line is latched on the rising edge of SCK. Data is output on the SO line by the falling edge of SCK.

Read Sequence

When reading from the E^{2}PROM array, $\overline{CS}$ is first pulled low to select the device. The 8-bit read instruction is transmitted to the X25128, followed by the 16-bit address of which the last 14 are used. After the read opcode and address are sent, the data stored in the memory at the selected address is shifted out on the SO line. The data stored in memory at the next address can be read sequentially by continuing to provide clock pulses. The address is automatically incremented to the next higher address after each byte of data is shifted out. When the highest address is reached ($3FFF) the address counter rolls over to address $0000 allowing the read cycle to be continued indefinitely. The read operation is terminated by taking $\overline{CS}$ high. Refer to the read E^{2}PROM array operation sequence illustrated in Figure 1.

To read the status register the $\overline{CS}$ line is first pulled low to select the device followed by the 8-bit instruction. After the RDSR opcode is sent, the contents of the status register are shifted out on the SO line. The read status register sequence is illustrated in Figure 2.

Write Sequence

Prior to any attempt to write data into the X25128, the "write enable" latch must first be set by issuing the WREN instruction (See Figure 3). $\overline{CS}$ is first taken low, then the WREN instruction is clocked into the X25128. After all eight bits of the instruction are transmitted, $\overline{CS}$ must then be taken high. If the user continues the write operation without taking $\overline{CS}$ high after issuing the WREN instruction, the write operation will be ignored.

To write data to the E^{2}PROM memory array, the user issues the write instruction, followed by the address and then the data to be written. This is minimally a thirty-two clock operation. $\overline{CS}$ must go low and remain low for the duration of the operation. The host may continue to write up to 32 bytes of data to the X25128. The only restriction is the 32 bytes must reside on the same page. If the address counter reaches the end of the page and the clock continues, the counter will "roll over" to the first address of the page and overwrite any data that may have been written.

For the write operation (byte or page write) to be completed, $\overline{CS}$ can only be brought high after bit 0 of data byte N is clocked in. If it is brought high at any other time the write operation will not be completed. Refer to Figures 4 and 5 below for a detailed illustration of the write sequences and time frames in which $\overline{CS}$ going high are valid.

To write to the status register, the WRSR instruction is followed by the data to be written. Data bits 0, 1, 4, 5 and 6 must be "0". This sequence is shown in Figure 6.

While the write is in progress, following a status register or E^{2}PROM write sequence, the status register may be read to check the WIP bit. During this time the WIP bit will be high.

Hold Operation

The $\overline{HOLD}$ input should be high (at V_{IH}) under normal operation. If a data transfer is to be interrupted $\overline{HOLD}$ can be pulled low to suspend the transfer until it can be resumed. The only restriction is the SCK input must

be low when $\overline{HOLD}$ is first pulled low and SCK must also be low when $\overline{HOLD}$ is released.

The $\overline{HOLD}$ input may be tied high either directly to V_{CC} or tied to V_{CC} through a resistor.

Operational Notes

The X25128 powers-up in the following state:

- The device is in the low power standby state.
- A high to low transition on $\overline{CS}$ is required to enter an active state and receive an instruction.
- SO pin is high impedance.
- The "write enable" latch is reset.

Data Protection

The following circuitry has been included to prevent inadvertent writes:

- The "write enable" latch is reset upon power-up.
- A WREN instruction must be issued to set the "write enable" latch.
- $\overline{CS}$ must come high at the proper clock count in order to start a write cycle.

Figure 1. Read E²PROM Array Operation Sequence

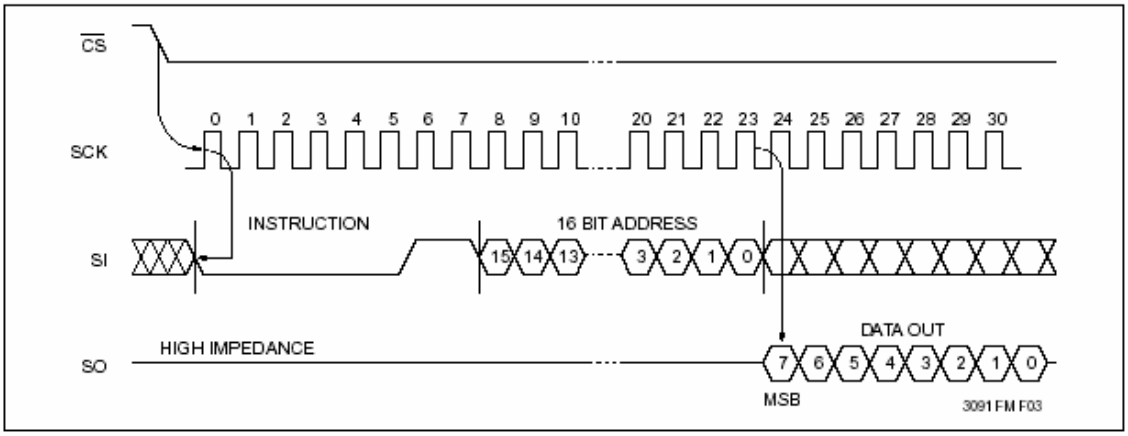


Figure 2. Read Status Register Operation Sequence

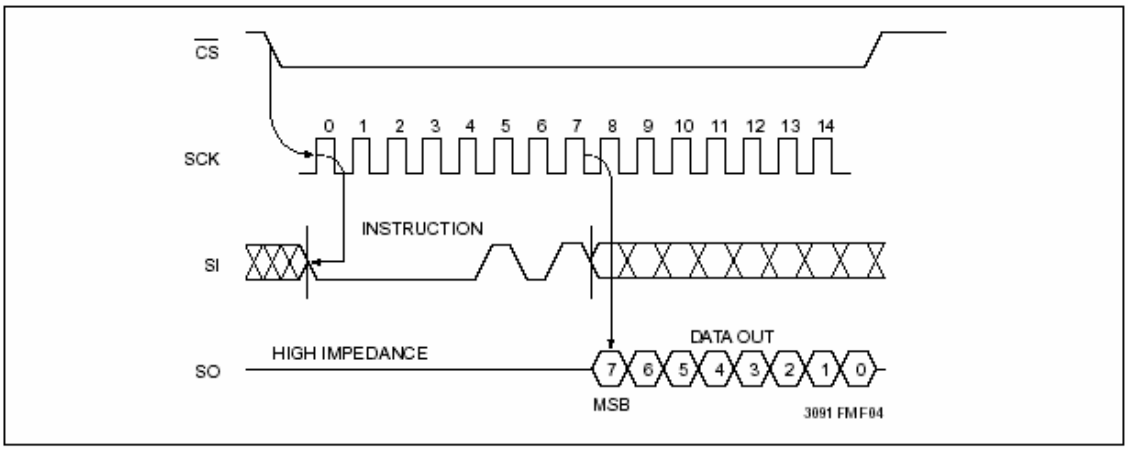


Figure 3. Write Enable Latch Sequence

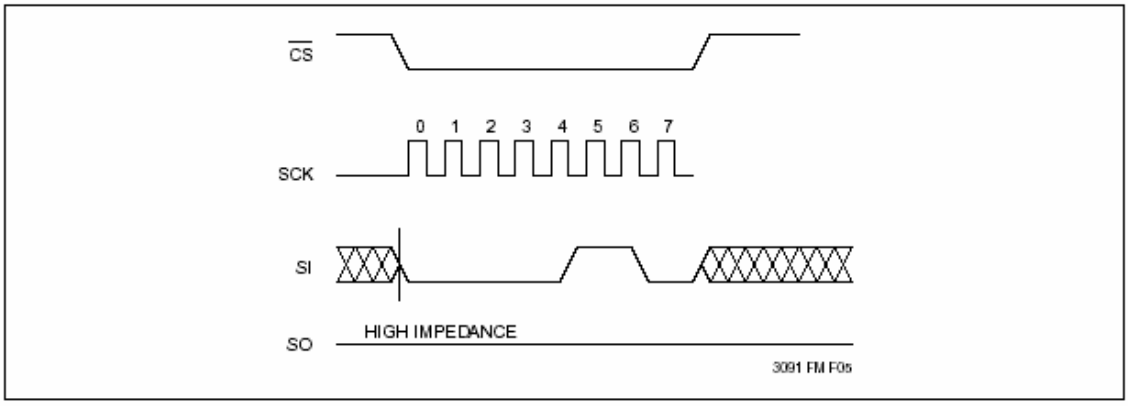


Figure 4. Byte Write Operation Sequence

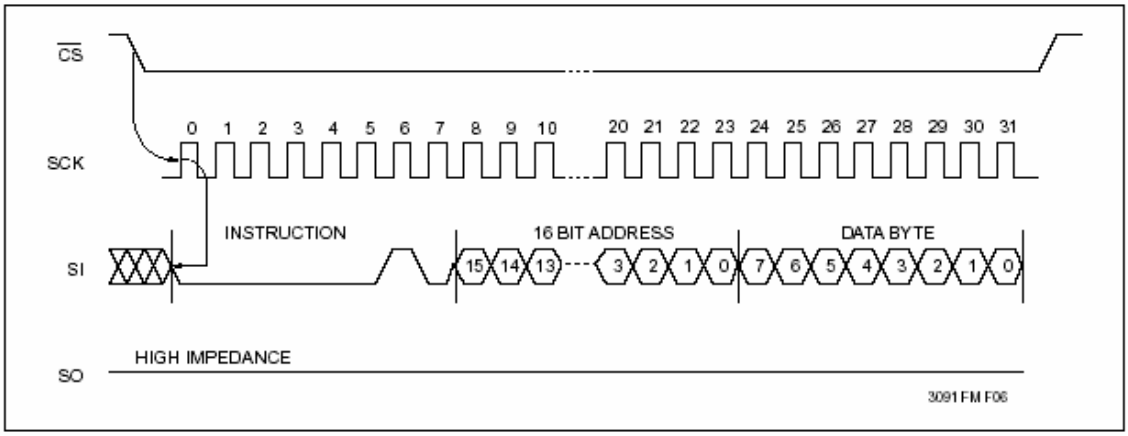


Figure 5. Page Write Operation Sequence

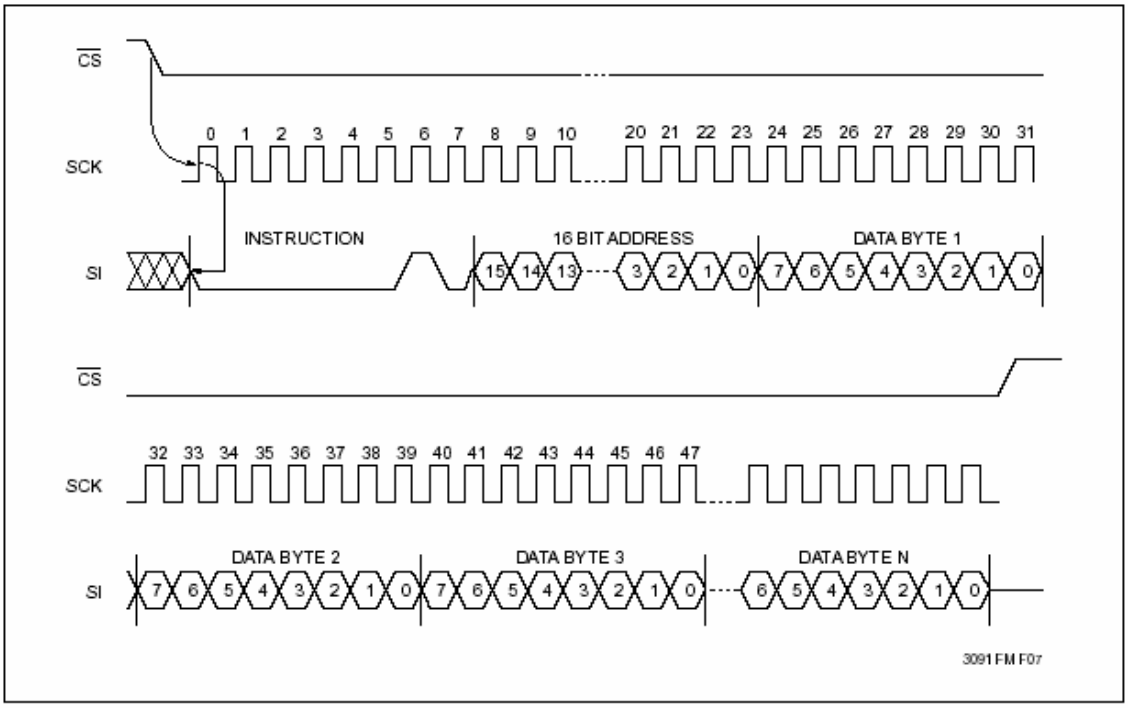
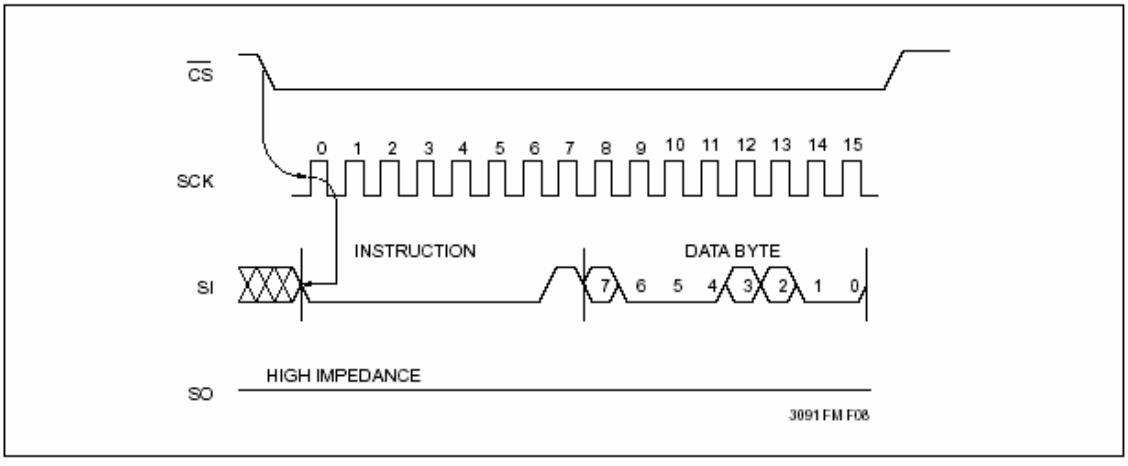


Figure 6. Write Status Register Operation Sequence



ABSOLUTE MAXIMUM RATINGS*

Temperature under Bias ..... –65°C to +135°C
Storage Temperature ..... –65°C to +150°C
Voltage on any Pin with
Respect to V_{SS} ..... –1V to +7V
D.C. Output Current 5mA
Lead Temperature
(Soldering, 10 seconds) ..... 300°C

*COMMENT

Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and the functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

RECOMMENDED OPERATING CONDITIONS

Temperature	Min.	Max.
Commercial	0°C	+70°C
Industrial	–40°C	+85°C
Military	–55°C	+125°C

3091 FMT06.1

Supply Voltage	Limits
X25128	5V ±10%
X25128-2.7	2.5V to 5.5V

3091 FMT07.2

D.C. OPERATING CHARACTERISTICS

Symbol	Parameter	Limits		Units	Test Conditions
		Min.	Max.		
I_{CC}	V_{CC} Supply Current (Active)		5	mA	SCK = V_{CC} x 0.1/V_{CC} x 0.9 @ 2 MHz, SO = Open, $\overline{CS}$ = V_{SS}
I_{SB}	V_{CC} Supply Current (Standby)		1	µA	$\overline{CS}$ = V_{CC}, V_{IN} = V_{SS} or V_{CC}
I_{LI}	Input Leakage Current		10	µA	V_{IN} = V_{SS} to V_{CC}
I_{LO}	Output Leakage Current		10	µA	V_{OUT} = V_{SS} to V_{CC}
V_{IL}(1)	Input LOW Voltage	–1	V_{CC} x 0.3	V	
V_{IH}(1)	Input HIGH Voltage	V_{CC} x 0.7	V_{CC} + 0.5	V	
V_{OL1}	Output LOW Voltage		0.4	V	V_{CC} = 5V, I_{OL} = 3mA
V_{OH1}	Output HIGH Voltage	V_{CC} – 0.8		V	V_{CC} = 5V, I_{OH} = -1.6mA
V_{OL2}	Output LOW Voltage		0.4	V	V_{CC} = 3V, I_{OL} = 1.5mA
V_{OH2}	Output HIGH Voltage	V_{CC} – 0.3		V	V_{CC} = 3V, I_{OH} = -0.4mA

POWER-UP TIMING

Symbol	Parameter	Min.	Max.	Units
t_{PUR}(3)	Power-up to Read Operation		1	ms
t_{PUW}(3)	Power-up to Write Operation		5	ms

3091 FMT09

CAPACITANCE T_A = +25°C, f = 1MHz, V_{CC} = 5V

Symbol	Test	Max.	Units	Conditions
C_{OUT}(2)	Output Capacitance (SO)	8	pF	V_{OUT} = 0V
C_{IN}(2)	Input Capacitance (SCK, SI, $\overline{CS}$, $\overline{WP}$, $\overline{HOLD}$)	6	pF	V_{IN} = 0V

3091 FMT10.1

Notes: (1) V_{IL} min. and V_{IH} max. are for reference only and are not tested.
(2) This parameter is periodically sampled and not 100% tested.
(3) t_{PUR} and t_{PUW} are the delays required from the time V_{CC} is stable until the specified operation can be initiated. These parameters are periodically sampled and not 100% tested.